

LOAD DISTRIBUTION METHOD FOR CLUSTER TYPE COMPUTER DEVICE

Patent number: JP9160885
Publication date: 1997-06-20
Inventor: MAYA YUZURU; UGAJIN ATSUSHI; KINOSHITA TOSHIYUKI
Applicant: HITACHI LTD
Classification:
- international: G06F15/16
- european:
Application number: JP19950316339 19951205
Priority number(s): JP19950316339 19951205

[View INPADOC patent family](#)

Abstract of JP9160885

PROBLEM TO BE SOLVED: To level the loads on processor modules and improve the throughput of the whole system by making a communication processor module calculate the process times of a transaction processor module and a file processor module and send a message to a reception-side processor module. **SOLUTION:** The function for the message process is divided into a communication process for a communication with a terminal, a transaction process, and a file process for controlling access to a disk. Transaction and file processor modules 14-16, and 17-19 send system operation information to communication processor modules 11-13 through a communication path 1 for control. The communication processor modules 11-13, on the other hand, calculate the process times of the reception modules and selects the processor modules having the shortest process time, thereby sending the message to the reception-side processor module.

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平9-160885

(43)公開日 平成9年(1997)6月20日

(51)Int.Cl.⁹

G 0 6 F 15/16

識別記号

3 8 0

庁内整理番号

F I

G 0 6 F 15/16

技術表示箇所

3 8 0 Z

審査請求 未請求 請求項の数20 O L (全 31 頁)

(21)出願番号 特願平7-316339

(22)出願日 平成7年(1995)12月5日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 真矢 譲

神奈川県川崎市麻生区王禅寺1099番地 株

式会社日立製作所システム開発研究所内

(72)発明者 宇賀神 敦

神奈川県海老名市今泉810番地 株式会社

日立製作所オフィスシステム事業部内

(72)発明者 木下 俊之

神奈川県川崎市麻生区王禅寺1099番地 株

式会社日立製作所システム開発研究所内

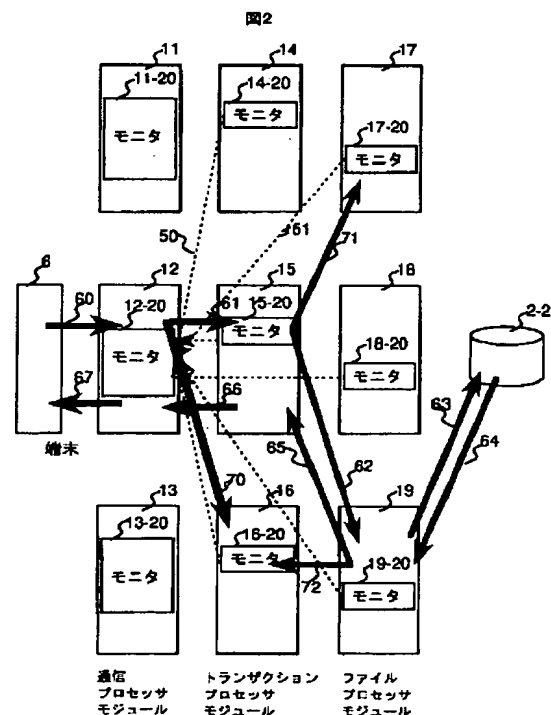
(74)代理人 弁理士 小川 勝男

(54)【発明の名称】 クラスタ型計算機装置の負荷分散方法

(57)【要約】

【課題】通信プロセッサモジュールは処理時間を算出し、受信側プロセッサモジュールに電文を送信することにより、処理能力を向上させ、障害時の回復時間を短縮する。

【解決手段】プロセッサモジュールは、プロセッサ、メモリおよびIOPから構成し、プロセス間通信バスと制御用通信バスにより接続する。電文処理は、端末との通信を行う通信プロセス、トランザクションプロセスおよびファイルプロセスに分割する。トランザクションとファイルプロセッサモジュール（通信プロセッサモジュール）は、制御用通信バスを介して、通信プロセッサモジュールにシステム稼働情報を送信する。一方、通信プロセッサモジュールは、電文を送信する際、受信側プロセッサモジュールの処理時間を計算し、処理時間より受信側プロセッサモジュールとその予備のプロセッサモジュールを選び、電文を送信する。



【特許請求の範囲】

【請求項1】それぞれがプロセッサ、メモリ、及びIOを制御するIOPからなる複数のプロセッサモジュールをプロセス間通信バスおよび制御用通信バスにより接続したシステムにおいて、

電文処理は、通信プロセス、トランザクションプロセスおよびファイルプロセスに分割してそれぞれを各プロセッサモジュールに搭載することによって通信プロセッサモジュール、トランザクションプロセッサモジュール、及びファイルプロセッサモジュールを構成し、前記トランザクションプロセッサモジュールおよび前記ファイルプロセッサモジュールは前記制御用通信バスを介して前記通信プロセッサモジュールにシステム稼働情報を送信し、通信プロセッサモジュールがシステム稼働情報を管理することを特徴とするクラスタ型計算機装置の負荷分散方法。

【請求項2】上記システムにおいて、前記トランザクションプロセッサモジュールと前記ファイルプロセッサモジュールは、システム立ち上げ時、あるいはプロセッサモジュール追加時に、自身のCPU処理能力、IO待ち時間およびプロセス間通信の実行ステップ数を前記通信プロセッサモジュールに送信することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項3】上記システムにおいて、前記プロセッサモジュールは多数の電文をいくつかのタイプに分け、電文のタイプ毎にシステム稼働情報として使用することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項4】上記システムにおいて、前記トランザクションプロセッサモジュールと前記ファイルプロセッサモジュールは、一定間隔毎に、システム稼働情報として、待ち行列に格納されている電文のタイプとその個数を、前記通信プロセッサモジュールに送信することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項5】上記システムにおいて、前記通信プロセッサモジュールは、受信した待ち行列に格納されている電文のタイプとその個数、CPU処理能力、IO待ち時間、プロセス間通信の実行ステップ数から、トランザクションプロセッサモジュールとファイルプロセッサモジュールからなる受信側プロセッサモジュールのすべてについて処理時間を算出し、処理時間が最小となるトランザクションプロセッサモジュールとファイルプロセッサモジュールを選び、前記通信プロセッサモジュールは、トランザクションプロセッサモジュールに電文と送信すべき前記ファイルプロセッサモジュールのアドレスを送信することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項6】上記システムにおいて、前記トランザクションプロセッサモジュールは、前記通信プロセッサモジ

ュールからの電文と前記ファイルプロセッサモジュールのアドレスを受信し、前記アドレスを参照して、前記ファイルプロセッサモジュールに電文を送信することの特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項7】上記システムにおいて、前記通信プロセッサモジュールはすべての前記トランザクションプロセッサモジュールと前記ファイルプロセッサモジュールの処理時間を算出し、前記処理時間の値が一定値以上ならば、輻輳状態と判定し、前記通信プロセッサモジュールは輻輳状態のプロセッサモジュールへの電文送信を禁止することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項8】上記システムにおいて、前記通信プロセッサモジュールが算出した処理時間が一定値以上のトランザクションプロセッサモジュールあるいはファイルプロセッサモジュールが輻輳状態に遷移した場合に、前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールは、前記通信プロセッサモジュールに対して電文の送信禁止を通知することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項9】上記システムにおいて、前記通信プロセッサモジュールはすべての前記トランザクションプロセッサモジュールと前記ファイルプロセッサモジュールの処理時間を算出し、前記処理時間の値が一定値以下ならば、輻輳状態から解除したと判定し、前記通信プロセッサモジュールは輻輳状態から解除した前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールへの電文送信を再開することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項10】上記システムにおいて、前記通信プロセッサモジュールは、輻輳状態の前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールの処理時間が一定値以下になった場合に、前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールは、前記通信プロセッサモジュールに対して電文の送信禁止を解除することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項11】上記システムにおいて、前記プロセッサモジュールが輻輳状態に遷移、あるいは輻輳状態から解除されたことを、前記制御用通信バスに接続されたコンソールに表示することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項12】上記システムにおいて、前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールで障害あるいは輻輳が発生した場合に、前記通信プロセッサモジュールは再度処理時間を計

算し、電文を再送することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項13】上記システムにおいて、前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールで障害あるいは輻輳状態から回復すると、前記通信プロセッサモジュールは再度処理時間を計算し、電文を送信することを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項14】上記システムにおいて、前記通信プロセッサモジュールは、処理時間が最小であって、電文を送信すべきトランザクションプロセッサモジュールとファイルプロセッサモジュールの次に処理時間の小さいトランザクションプロセッサモジュールとファイルプロセッサモジュールを予備のプロセッサモジュールとして稼働させて電文を送信し、障害が発生すると、前記予備のプロセッサモジュールが処理を引き継ぐことを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項15】上記システムにおいて、前記トランザクションプロセッサモジュールあるいはファイルプロセッサモジュールで障害が発生した場合に、前記予備のトランザクションプロセッサモジュールあるいはファイルプロセッサモジュールは、自身が受信している電文を参照することにより、前記通信プロセッサモジュールからの電文の再送を要求しないで引き継ぎ処理を実行することを特徴とする請求項14記載のクラスタ型計算機装置の負荷分散方法。

【請求項16】上記システムにおいて、障害から回復すると、前記トランザクションプロセッサモジュールあるいは前記ファイルプロセッサモジュールは前記通信プロセッサモジュールに障害回復を通知し、処理時間を算出することを特徴とする請求項14記載のクラスタ型計算機装置の負荷分散方法。

【請求項17】上記システムにおいて、前記プロセッサモジュールのそれぞれは複数のプロセスを搭載し、1つのプロセスを現用プロセスとして稼働させ、他のプロセスを予備プロセスとして待機させ、あるプロセッサモジュールで障害が発生した場合に、障害プロセッサモジュールを閉塞し、さらに更にシステム処理能力が最高になるように、前記障害プロセスを他の機能のプロセッサモジュールが引き継ぐことを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項18】上記システムにおいて、電文をn個のプロセスに分割したn階層のクラスタ型計算機を構成し、最上位プロセッサモジュールが下位のプロセッサモジュールの処理時間を算出し、これが最小となるプロセッサモジュールで電文処理を実行させることを特徴とする請求項1記載のクラスタ型計算機装置の負荷分散方法。

【請求項19】上記システムにおいて、前記最上位プロセッサモジュールが下位のプロセッサモジュールの処理時間を算出し、前記処理時間が2番目に小さいプロセッ

サモジュールを予備のプロセッサモジュールとして稼働させ、前記予備のプロセッサモジュールにも電文を送信し、障害時および輻輳時に前記予備のプロセッサモジュールが処理を引き継ぐことを特徴とする請求項16記載のクラスタ型計算機装置の負荷分散方法。

【請求項20】上記システムにおいて、前記最上位プロセッサモジュールが前記通信プロセッサモジュールの機能を実行することを特徴とする請求項17記載のクラスタ型計算機装置の負荷分散方法。

【発明の詳細な説明】

【0001】

【発明に属する技術分野】本発明は、複数のプロセッサモジュールをプロセス間通信バスおよび制御用通信バスにより接続し、電文処理を複数のプロセスに分割し、分割されたプロセスを複数のプロセッサモジュールに搭載したクラスタ型計算機システムに係わり、プロセッサモジュールの負荷を均一にすることにより、システム全体の処理能力を向上させ、更に障害時におけるシステム回復時間を短縮するクラスタ型計算機装置およびその負荷分散方法に関する。

【0002】

【従来の技術】計算機システムはダウンサイジングにより、汎用大型機を中心としたシステムから多数のプロセッサモジュールを高速バスにより接続する分散システムに移行している。一方、計算機システムの処理形態はオンライントランザクション処理が主流になり、年々トランザクション処理量が増大し、高い処理能力が要求されている。

【0003】ところで、従来の分散機では、日経エレクトロニクス(1992.5.18、No.554、p.87～p.96)に記載されている。これは、現用系と待機系からなるホットスタンバイ構成であり、これらの系間に制御用通信バスを設け、障害を検出するためのaliveメッセージのみを送信していた。しかし、このようなホットスタンバイシステムでは電文の処理時間やI/O(入出力)待ち時間のようなシステム稼働情報を他系に送信していなかった。

【0004】一方、従来の負荷分散方法は、“マルチプロセッサ時分割システムにおける負荷分散アルゴリズム”(電子情報通信学会論文誌 D-I Vol. J72-D-I No.2 PP.75-82 1989年2月)に記載されているように、2つのプロセッサモジュール間でCPU利用率あるいは応答時間のようなシステム稼働情報により、プロセッサモジュール間の負荷分散を行っていた。

【0005】最後に、並列計算機でも同様の技術が行われており、「並列計算機構成論」、富田真治著(p.195～p.200)では、以下のように記載されている。すなわち、並列計算機では、フェールセーフシステムの適用により、1プロセッサに複数の機能を持たせない。プロセッサ障害時には、処理を引き継ぐのではなく、障害プロセッサを閉塞させている。

【0006】SMS-201の場合、動作モードは制御・通信および実行の各モードに分割されている。制御・通信モードでは、マスタプロセッサ（メインプロセッサ）は各プロセッサの結果が格納されている通信用メモリの出力バッファの内容を読み出し、それをすべてのプロセッサの通信用メモリの入力バッファにブロードキャスト方式により書き込む。従って、各プロセッサの通信用メモリの内容はすべて同一になる。実行モードでは、各プロセッサは通信メモリを使用して、相互に全く独立に動作し、結果を通信用メモリの出力バッファに格納し、処理の終了を主プロセッサに報告し、制御・通信モードを待つ。

【0007】

【発明が解決しようとする課題】クラスタ型計算機システムにおいて、オンライントランザクション処理を実行する場合には、複数のプロセッサモジュールをプロセス間通信バスおよび制御用通信バスにより接続し、電文処理を複数のプロセス（通信プロセス、トランザクションプロセスおよびファイルプロセス）に分割する。そして、分割されたプロセスはそれぞれ多数のプロセッサモジュールで実行される。

【0008】そして、通信プロセッサモジュールが端末から電文を受信すると、トランザクションプロセッサモジュールに電文を送信する。そして、トランザクションプロセッサモジュールは電文を解析し、ファイルプロセッサモジュールにディスクの読み出し処理や書き込み処理を要求する。更に、ファイルプロセッサモジュールはディスクへの読み出し処理や書き込み処理を実行する。その後、ファイルプロセッサモジュールは、これらの処理が終了すると、ディスクのアクセス要求の送信元のトランザクションプロセッサモジュールにディスクへの書き込み完了を通知する。そして、トランザクションプロセッサモジュールは、電文送信元の通信プロセッサモジュールに処理結果を送信する。最後に、通信プロセッサモジュールは処理結果を端末に通知する。

【0009】このように、トランザクション処理では、ディスクの参照および更新に関し、通信プロセッサモジュールからファイルプロセッサモジュールまでの処理の経路では、行きと帰りの経路は同一のプロセッサモジュールでなければならない。

【0010】ところで、従来の分散機構では、現用系と待機系からなるホットスタンバイ構成であり、これらの系間に制御用通信バスを設け、障害を検出するためのaliveメッセージのみを送信していた。しかし、このようなホットスタンバイシステムでは電文の処理時間やIO待ち時間のようなシステム稼働情報を他系に送信していなかった。その結果、負荷の分散は考慮されていなかった。

【0011】次に、従来の負荷分散方法は、“マルチプロセッサ時分割システムにおける負荷分散アルゴリズム”

（電子情報通信学会論文誌 D-I Vol. J72-D-I No.2 PP.75-82 1989年2月）に記載されているように、2つのプロセッサモジュール間でCPU利用率あるいは応答時間のようなシステム稼働情報により、プロセッサモジュール間の負荷分散を行っていた。このシステムでは、マスタプロセッサを設けていないため、2つのプロセッサモジュール間で負荷分散は可能であるが、システム全体を考慮した負荷分散は行えないという問題があった。

【0012】ところで、従来の負荷分散方法では、マスタプロセッサを設けておらず、通信プロセッサとトランザクションプロセッサ間、およびトランザクションプロセッサとファイルプロセッサ間で負荷分散を行っており、システム全体での負荷分散を行っていなかった。

【0013】最後に、並列計算機でも同様の技術が行われており、「並列計算機構成論」、富田真治著（p.195～p.200）では、以下のように記載されている。並列計算機では、フェールセーフシステムの適用により、1プロセッサに複数の機能を持たせない。プロセッサ障害時には、処理を引き継ぐのではなく、障害プロセッサを閉塞させている。SMS-201の場合には、マスタプロセッサを設けているが、負荷分散制御は行っていない。また、各プロセッサモジュールは、単一の機能のみを実行している。その結果、並列計算機では、プロセッサモジュール障害が発生すると、特定のプロセスの処理能力が低下し、システム全体の処理能力が低下する。更に、待機プロセスを設けていないため、予備のルートを設定することができなくなり、障害時におけるシステム回復時間が長くなるという問題がある。

【0014】また、プロセッサモジュールで障害が発生すると、チェックポイントデータの取得により、実行中の電文の引き継ぎは可能であるが、入力待ち行列に格納されている電文は消滅する。このため、電文の再送処理が必要となり、回復時間が長くなるという問題が残る。

【0015】本発明の目的は、クラスタ型計算機システムにおいて、通信プロセッサモジュールが他のすべてのプロセッサモジュールを管理し、プロセッサモジュール間の負荷を均一にし、システム全体の処理能力を向上させることにある。また、本発明の他の目的は、受信側プロセッサモジュールの処理時間により予備のプロセッサモジュールを選んで電文を送信させておき、受信側プロセッサモジュールで障害が発生しても、電文の再送処理をなくし、回復時間を短縮させることにある。

【0016】

【課題を解決するための手段】上記のようにプロセスを分割したシステムの処理能力は、プロセス毎に処理能力を算出し、これらのうち最も低いプロセスの処理能力になる。このため、同一プロセスを実行しているプロセッサモジュール間に負荷のばらつきがあると、システム処理能力の低下の要因となる。

【0017】本発明の目的を達成するために、すべての

プロセッサモジュールを制御用通信バスとプロセス間通信バスにより接続し、受信側プロセッサモジュール（以下、トランザクションプロセッサモジュールとファイルプロセッサモジュール）は、電文の実行時間等のシステム稼働情報を制御用通信バスを介して、通信プロセッサモジュールに送信する。一方、通信プロセッサモジュールはトランザクションプロセッサモジュールとファイルプロセッサモジュールの処理時間を算出し、処理時間が最小となるトランザクションプロセッサモジュールとファイルプロセッサモジュールを選択する。これにより、通信プロセッサモジュールはこれらの受信側プロセッサモジュールに電文を送信することにより、プロセッサモジュールの負荷を均一にし、システム全体の処理能力を向上させる。

【0018】本発明の他の目的を達成するために、上記で算出した処理時間から、トランザクションプロセッサモジュールとファイルプロセッサモジュールはそれぞれ予備のプロセッサモジュールを選び、予備のプロセッサモジュールにも電文を送信し、実際に電文処理を行っているプロセッサモジュールに障害が発生しても、回復時間を短縮させることができる。

【0019】電文の受信側プロセッサモジュールは、制御用通信バスを介して、一定間隔毎に通信プロセッサモジュールにシステム稼働情報として、待ち行列に格納されている電文のタイプとその個数を送信する。一方、受信側プロセッサモジュールは、システム立ち上げ時に、通信プロセッサモジュールに処理能力や送信する電文の処理ステップ数の情報を送信しておく。そして、通信プロセッサモジュールは、電文を送信する場合に受信側プロセッサモジュールの処理時間を計算する。そして、処理時間の最も短いトランザクションプロセッサモジュールとファイルプロセッサモジュールとこれらの予備のプロセッサモジュールを選び、これらのトランザクションプロセッサモジュールとファイルプロセッサモジュールに電文を送信する。障害が発生すると、予備のプロセッサモジュールが引き継ぎ処理を実行する。

【0020】本発明では、プロセッサモジュールは、プロセッサ、メモリ、IOPおよびIO制御装置から構成する。複数のプロセッサモジュールは、プロセス間通信バスと制御用通信バスにより接続される。電文処理の機能は、端末との通信を行う通信プロセス、トランザクションプロセスおよびディスクとのアクセスを制御するファイルプロセスに分割される。

【0021】まず、トランザクションプロセッサモジュールおよびファイルプロセッサモジュールは、制御用通信バスを介して、待ち行列に格納されている電文のタイプとその個数をシステム稼働情報として、通信プロセッサモジュールに送信する。

【0022】一方、通信プロセッサモジュールは、受信した待ち行列に格納されている電文のタイプとその個

数、およびあらかじめ受信側プロセッサモジュールから受信しておいた受信側プロセッサモジュールの処理能力や電文の処理ステップ数の情報から処理時間を計算する。そして、通信プロセッサモジュールは処理時間が最も短いプロセッサモジュールを選び、選んだ受信側プロセッサモジュールに電文を送信する。同様に、予備となるプロセッサモジュールを選び、これらのプロセッサモジュールに電文を送信しておく。

【0023】このようにして、通信プロセッサモジュールは、受信側プロセッサモジュールから得た情報に基づいて、各受信側プロセッサモジュールの負荷を均一とし、システム全体の処理能力を向上できる。また、プロセッサモジュールで障害が発生しても、予備のプロセッサモジュールが電文を受信しているため、回復時間を短縮できる。

【0024】

【発明の実施の形態】以下、本発明の実施例を示す。図1は、本発明によるシステム構成図である。本発明によるシステム構成は、9個のプロセッサモジュール（11～19）を前提とする。すべてのプロセッサモジュール（11～19）は、制御用通信バス（1）およびプロセス間通信バス（3）により接続する。

【0025】制御用通信バス（1）は、システム稼働情報、リセットメッセージおよびaliveメッセージを通信する。リセットメッセージは障害の発生したプロセッサモジュール（11～19）へ閉塞を要求するメッセージである。aliveメッセージはプロセッサモジュール（11～19）が正常に動作しているかどうかチェックするメッセージである。システム稼働情報は、本発明で使用するメッセージであり、プロセッサモジュール（11～19）の実行中の情報を示すものである。

【0026】プロセス間通信バス（3）は、プロセス間で電文を通信するために使用する。プロセッサモジュール（11～13）は、図5で説明するように通信プロセス（30）を搭載する。このため、プロセッサモジュール（11～13）は業務用LAN（4）および回線切替装置（5）と接続し、端末（6-1～4）と通信できるようにする。また、プロセッサモジュール（11～13）は受信電文の識別が必須であるため、電文に時刻印を付与する。このために、プロセッサモジュール（11～13）はタイマ（7）と接続されている。プロセッサモジュール（11～13）は、共有ディスク（2-0）を共有し、障害発生時に、処理を引き継ぐために必要なチェックポイント情報を格納する。

【0027】一方、他のプロセッサモジュール（14～19）は、通信プロセス（30）を搭載しないため、業務用LAN（4）あるいは回線切替装置（5）と接続されていない。プロセッサモジュール（14～16）は共有ディスク（2-1）を共有し、プロセッサモジュール（14～16）の障害発生時に、処理を引き継ぐために

必須なチェックポイント情報を格納する。プロセッサモジュール(17~19)は共有ディスク(2-2)を共有し、プロセッサモジュール(17~19)の障害発生時に、処理を引き継ぐために必須なチェックポイント情報を格納する。また、共有ディスク(2-2)には、端末(6)が参照あるいは更新するデータを格納する。

【0028】図2は本発明の処理概要を示す図である。トランザクションプロセッサモジュール(14~16)は、一定間隔毎にシステム稼働情報を通信プロセッサモジュール(11~13)に送信する(処理50)。同様に、ファイルプロセッサモジュール(17~19)は、一定間隔毎にシステム稼働情報を通信プロセッサモジュール(11~13)に送信する(処理51)。

【0029】一方、通信プロセッサモジュール(11~13)は、トランザクションプロセッサモジュール(14~16)とファイルプロセッサモジュール(17~19)のすべての組合せにおいて、処理時間を算出し、これが最小となるように電文を送信するトランザクションプロセッサモジュール(15)とファイルプロセッサモジュール(19)を決定する。

【0030】次に、プロセッサモジュール(11~19)の処理手順について説明する。通信プロセッサモジュール(12)は端末(6)から電文を受信する(処理60)。そして、通信プロセッサモジュール(12)は、処理時間が最小となるトランザクションプロセッサモジュール(15)に電文を送信する。更に、トランザクションプロセッサモジュール(15)は、通信プロセッサモジュール(12)から受信した処理時間が最小となるファイルプロセッサモジュール(19)に電文を送信する(処理62)。

【0031】そして、ファイルプロセッサモジュール(19)はディスクの読み出し処理(処理63)および書き込み処理を実行する(処理64)。そして、その処理結果をトランザクションプロセッサモジュール(15)に戻す(処理65)。更に、トランザクションプロセッサモジュール(15)は、処理結果を通信プロセッサモジュール(12)に戻す(処理66)。最後に、通信プロセッサモジュール(12)は処理結果を端末(6)に戻す(処理67)。

【0032】このように、通信プロセッサモジュール(12)は、受信側プロセッサモジュール(14~19)から常に電文のタイプとその個数を受信しておき、通信プロセッサモジュール(11~13)は、受信側プロセッサモジュール(14~19)の処理能力および電文の処理ステップ数により、処理時間を計算し、処理時間が最小となるプロセッサモジュール(14~19)に電文を送信する。これにより、通信プロセッサモジュール(12)は受信側プロセッサモジュール(14~19)の負荷を均一にし、システム全体の処理能力を向上させることが可能となる。

【0033】更に、通信プロセッサモジュール(12)は処理時間により、予備のプロセッサモジュール(16、17)を選び、予備のプロセッサモジュール(16、17)は電文受信処理(処理70~72)のみ実行することにより、障害時に電文の再送処理をなくすることが可能となり、回復時間を短縮できる。

【0034】図3は、プロセッサモジュールの構成図である。プロセッサモジュール(11~19)は業務用LAN(4)あるいは回線切替装置(5)との接続を除いて、同一構成であるため、プロセッサモジュール(11)を例にして説明する。プロセッサモジュール(11)はプロセッサ(11-1)、メモリ(11-2)、IOP(11-3)、回線制御装置(11-4)、プロセス間通信バス制御装置(11-5)、ディスク制御装置(11-6)、制御用通信バス制御装置(11-7)、業務用LAN制御装置(11-8)、およびシステム監視装置(11-9)から構成される。プロセッサモジュール(12、13)は、プロセッサモジュール(11)と同様に通信プロセス(30)を搭載するため、同一の構成である。なお、回線制御装置(11-4)、プロセス間通信バス制御装置(11-5)、ディスク制御装置(11-6)、制御用通信バス制御装置(11-7)、および業務用LAN制御装置(11-8)を合わせて、IO制御装置(11-10)という。

【0035】一方、プロセッサモジュール(14~19)は、業務用LAN(4)あるいは回線切替装置(5)とは接続されないため、これらの制御装置は不要となる。プロセッサモジュール(14~19)は、プロセッサ(14-1~19-1)、メモリ(14-2~19-2)、IOP(14-3~19-3)、プロセス間通信バス制御装置(14-5~19-5)、ディスク制御装置(14-6~19-6)、制御用通信バス制御装置(14-7~19-7)、およびシステム監視装置(14-9~19-9)から構成される。

【0036】図4は電文処理の概要を示す図である。電文処理は、端末(6)から電文を受信する。そして、ディスク装置(2-2)から必要なデータを読み出す。電文処理を実行し、処理結果をディスク装置(2-2)に書き込む。そして、端末(6)に応答を返す。このため、本実施例では、電文処理は、端末(6)との通信を行う通信プロセス(30)、トランザクションプロセス(31)、およびディスク装置(2-2)のアクセスを制御するファイルプロセス(32)に分割される。

【0037】図5は、ソフトウェアの構成と電文処理の分割を示す図である。プロセッサモジュール(11~19)のソフトウェアは、それぞれモニタ(11-20~19-20)、OS(11-21~19-21)、およびプロセス(30~32)から構成される。モニタ(11-20~19-20)は、プロセッサモジュール間通信情報を管理する。

【0038】電文処理の機能は、通信プロセス(30)、トランザクションプロセス(31)、およびファイルプロセス(32)に分割される。そして、通信プロセス(30)はプロセッサモジュール(11~13)に、トランザクションプロセス(31)はプロセッサモジュール(14~16)に、ファイルプロセス(32)はプロセッサモジュール(17~19)にそれぞれ搭載される。

【0039】以下、トランザクションプロセッサモジュール(14~16)とファイルプロセッサモジュール(17~19)を合わせて、受信側プロセッサモジュール(14~19)という。

【0040】図6は、I/O制御装置のシステム構成図である。ここでは、回線制御装置(11-4)を例にして、説明する。回線制御装置(11-4)は、プロセッサ(11-4-1)、メモリ(11-4-2)、バッファ(11-4-3)、ディスク制御部(11-4-4)で構成される。バッファ(11-4-3)には、他のプロセッサモジュール(14~16)から受信した電文を格納する入力待ち行列として、入力待ち行列20(11-4-5)と入力待ち行列21(11-4-7)を設け、また他のプロセッサモジュール(14~16)に送信する電文を格納する出力待ち行列として、出力待ち行列20(11-4-6)と出力待ち行列21(11-4-8)を設ける。

【0041】ここで、端末(6)からディスク(2-2)への情報の流れに対応して、上位のプロセッサモジュールおよび下位のプロセッサモジュールを、以下のよう

に定義する。
【0042】自身のプロセッサモジュールからディスク(2-2)側のプロセッサモジュールを下位のプロセッサモジュールとする。自身のプロセッサモジュールから端末(6)側のプロセッサモジュールを上位のプロセッサモジュールとする。例えば、トランザクションプロセッサモジュール(14~16)から見ると、下位のプロセッサモジュールはファイルプロセッサモジュール(17~19)であり、上位のプロセッサモジュールは通信プロセッサモジュール(11~13)である。

【0043】I/O制御装置(11-10)の待ち行列について、入力待ち行列20(11-4-5)と出力待ち行列20(11-4-6)は上位のプロセッサモジュール用であり、入力待ち行列21(11-4-7)と出力待ち行列21(11-4-8)は下位のプロセッサモジュール用である。

【0044】I/O制御装置(11-10)では4つの待ち行列があるが、システム稼働情報は、上位および下位のプロセッサモジュールから受信している電文の情報であるため、入力待ち行列20(11-4-5)と入力待ち行列21(11-4-7)を対象とする。

【0045】図7はプロセッサモジュールの状態遷移図

である。ここでは、現用状態(150)、輻輳状態(151)、およびオフライン状態(152)の3つの状態を設ける。現用状態(150)とは、正常に電文処理を実行している状態である。輻輳状態(151)は、一時的に処理能力の以上の電文を受信し、他のプロセッサモジュール(11~19)からの電文受信を禁止している状態である。オフライン状態(152)は障害が発生している状態である。

【0046】現用状態(150)で処理を実行中に、処理時間が一定値以上になると、輻輳状態(151)に遷移する(154)。輻輳状態(151)の場合に、処理時間が一定値以下になると、現用状態(150)に遷移する(155)。現用状態(150)あるいは輻輳状態(151)で障害が発生すると、オフライン状態(152)に遷移する(156、157)。その後、障害から回復すると、オフライン状態(152)から現用状態(150)に遷移する(158)。

【0047】図8は、16進コードで表わしたプロセッサモジュールの状態コードを示す図である。現用状態(150)は(01)16と、輻輳状態(151)は(02)16と、オフライン状態(152)は(03)16とする。

【0048】図9は制御用通信パスにおける電文フォーマットを示す図である。電文フォーマットは、送信元アドレス(80)、送信先アドレス(81)、制御用電文種別(82)、システム稼働情報1(83)、およびシステム稼働情報2(84)から構成される。

【0049】以下、制御用通信パス(1)における電文情報について詳細に説明する。図10は、16進コードで表わした制御用電文種別コードを示す図である。制御用電文(82)には、リセットメッセージ、aliveメッセージ、およびシステム稼働情報がある。これらのコードは図10に示すように、それぞれ(10)16、(01)16および(02)16とする。

【0050】図11は、システム稼働情報1とシステム稼働情報2の組合せと、対応する意味を示す図である。システム稼働情報1(83)が(01)16の場合には、入力待ち行列20と入力待ち行列21に格納されている電文のタイプとその個数を示す。

【0051】システム稼働情報1(83)が(10)16の場合には、プロセッサモジュール(11~19)あるいはコンソール(8)に転送するイベント情報を示す。さらに、システム稼働情報2(84)が(01)16の場合には1つのプロセッサモジュールで輻輳発生を、(02)16の場合には同一プロセスの全プロセッサモジュールで輻輳発生を、(03)16の場合には障害発生を、(10)16の場合には輻輳から回復したことを、(11)16の場合には障害から回復したことをそれぞれ示す。

【0052】図12は電文のタイプとその個数を示す図である。入力待ち行列20に格納されている電文は、電

文タイプ(84-1-0~84-5-0)とその個数(84-1-1~84-1-5)であり、入力待ち行列21に格納されている電文は、電文タイプ(84-1-2~84-5-2)とその個数(84-1-3~84-1-3)である。

【0053】図13は電文のタイプとそのコードの対応を示す図である。ここでは、電文1は(01)16とし、以下順にコードを設定し、最後の電文5は(05)16とする。通常のデータは、図13に示した電文の授受の後に送受信される。従って、以下の説明では、プロセスの中での通常のデータの送受信については省略する。

【0054】図14はプロセッサモジュールとコンソールのそれぞれのアドレスを示す図である。このアドレスは、プロセス間通信バス(3)と制御用通信バス(1)ともに、同じアドレスを使用する。プロセッサモジュール(11)のアドレスは(00)16とする。以下、プロセッサモジュール(12~19)とコンソール(8)のアドレスを図14に示す通りに設定する。

【0055】図15はプロセス間通信の電文フォーマットを示す図である。電文フォーマットは、送信元アドレス(90)、送信先アドレス(91)、電文本体(92)、下位のプロセッサモジュールのアドレス(93)、および識別子(94)から構成される。

【0056】図16はトランザクションプロセッサモジュールの入力待ち行列に格納されている電文を示す図である。3つのトランザクションプロセッサモジュール(14~16)の入力待ち行列20(14-4-5~16-4-5)には、それぞれ5個、2個、3個の電文が格納されており、入力待ち行列21(14-4-7~16-4-7)にはそれぞれ1個、0個、0個の電文が格納されている。これらの電文のタイプは図16に示す通りである。

【0057】図17はファイルプロセッサモジュールの入力待ち行列に格納されている電文を示す図である。ファイルプロセッサモジュール(17~19)の入力待ち行列20(17-4-5~19-4-5)には、それぞれ3個、0個、2個の電文が格納されている。電文のタイプは図17に示す通りである。ファイルプロセッサモジュールは最下位のプロセッサモジュールであるため、入力待ち行列21(17-4-7~19-4-7)は使用しない。

【0058】図18は、トランザクションプロセッサモジュールの電文毎の処理ステップ数と、ファイルプロセッサモジュールとの通信回数を示す図である。ここでは、電文1から電文5におけるトランザクションプロセッサモジュール(14~16)の実行ステップ数(上位のプロセッサモジュール用のST1(i)と下位のプロセッサモジュール用のST2(i))と、ファイルプロセッサモジュール(17~19)および通信プロセッサモジュール(11~13)との通信回数(CT1(i)とCT2(i))を示

す。

【0059】これらの情報は、すべての通信プロセッサモジュール(11~13)が所有し、処理時間を算出するために使用する。ここでは、通信プロセッサモジュール(11~13)およびファイルプロセッサモジュール(17~19)から受信した電文毎の処理ステップ数を示す。

【0060】図19はファイルプロセッサモジュールの電文毎の処理ステップ数とI/O発行回数を示す図である。ここでは、電文1~電文5におけるファイルプロセッサモジュール(17~19)の実行ステップ数(SF1(i))、通信回数(CF(i))、およびI/O回数(IF1(i))を示す。これらの情報は、すべての通信プロセッサモジュール(11~13)が所有し、処理時間を算出するために使用する。

【0061】図20は、各トランザクションプロセッサモジュールの処理能力と状態を示す図である。図21は、各ファイルプロセッサモジュールの処理能力と状態を示す図である。

【0062】図20と図21について、処理能力は1秒当りの実行ステップ数を表わす。また、プロセッサモジュールの状態は図7に示したものである。ここでは、プロセッサモジュール(14)は輻輳状態(151)とし、プロセッサモジュール(18)は障害が発生しオフライン状態(152)とする。他のプロセッサモジュール(15~17、19)は正常に動作しており、現用状態(150)とする。

【0063】これ以降、各プロセッサモジュール(11~19)の処理手順について説明する。最初に、図22から図24より、システム稼働情報の送受信処理について説明する。

【0064】図22はシステム稼働情報による処理手順を示す図である。図23は通信プロセッサモジュールの処理時間の算出式である。この算出式より、トランザクションプロセッサモジュール(14~16)の処理時間を算出する。トランザクションプロセッサモジュール(14)は図20より輻輳状態(151)である。処理時間は、トランザクションプロセッサモジュール(15)が最小となり、トランザクションプロセッサモジュール(16)が2番目に小さい。トランザクションプロセッサモジュール(15)がトランザクションプロセスの処理を実行し、トランザクションプロセッサモジュール(16)を予備のプロセッサモジュールとする。

【0065】同様に、ファイルプロセッサモジュール(17~19)の処理時間を算出する。ファイルプロセッサモジュール(18)は図21よりオフライン状態である。処理時間は、ファイルプロセッサモジュール(19)が最小となり、ファイルプロセッサモジュール(17)が2番目に小さい。ファイルプロセッサモジュール(19)がトランザクションプロセスの処理を実行し、

ファイルプロセッサモジュール(17)を予備のプロセッサモジュールとする。

【0066】図24は制御用電文の一例を示す図である。図24(1)は待ち行列に格納されている電文の情報の転送を示す。図24(2)は、上位のプロセッサモジュールへの電文送信禁止の通知を示す。図24

(3)、(4)、(5)は、それぞれケース1からケース3のコンソール(8)への表示を示す。

【0067】ここでは、ファイルプロセッサモジュール(19)が通信プロセッサモジュール(12)にシステム稼働情報を送信する場合を示す。まず、ファイルプロセッサモジュール(19)は、通信プロセッサモジュール(12)に対して、システム稼働情報として、入力待ち行列20(19-4-7)に格納されている電文のタイプとその個数を送信する(処理400)。

【0068】この電文を図24(1)に示す。ここでは、ファイルプロセッサモジュールはプロセッサモジュール(19)とする。送信元アドレス(80)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(81)はプロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)は電文のタイプと個数であるため(01)16を、それぞれ設定する。また、システム稼働情報2(84)は電文のタイプ毎にその個数を格納するため、図17に示している通り、ファイルプロセッサモジュール(19)には電文5が1個と電文2が1個格納されているため、電文のタイプ毎に、電文のタイプの順に、電文のタイプとその個数を、電文1では0個であるため、(0100)16、電文2では、(0201)16、電文3では(0300)16、電文4では(0400)16を、電文5では(0501)16を設定する。ファイルプロセッサモジュール(19)は入力待ち行列21(19)を使用しないため、入力待ち行列21用(84-1-2~84-5-3)には(F)16を格納する。

【0069】通信プロセッサモジュール(12)は、ファイルプロセッサモジュール(19)の入力待ち行列20(19-4-5)と入力待ち行列21(19-4-7)に格納されている電文タイプとその個数を受信する(処理401)。そして、通信プロセッサモジュール(12)は、ファイルプロセッサモジュール(19)の処理時間を算出する(処理402)。通信プロセッサモジュールの処理時間の算出式を図23に示す。

【0070】ここで、通信プロセッサモジュール(12)は、受信側プロセッサモジュール(14~19)の電文の実行ステップ数、通信回数、およびIO待ち時間を事前に格納しておく。この例を図18と図19に示す。さらに、受信側プロセッサモジュール(14~19)の処理能力とプロセッサモジュールの状態を事前に格納しておく。この例を図20と図21に示す。

【0071】図23は処理時間の算出式である。トランザクションプロセッサモジュール(14~16)の処理時間(RT(j))は、電文タイプ毎の実行ステップ数(ST1(i)とST2(i))と電文数(NT1(i)とNT2(i))、およびプロセス間通信回数(CT1(i)とCT2(i))とプロセス間通信のステップ数(PS)から算出する。

【0072】ファイルプロセッサモジュール(17~19)の処理時間(RF(j))は、電文タイプ毎の実行ステップ数(SF1(i))と電文数(NF1(i))、プロセス間通信回数(CF1(i))とプロセス間通信のステップ数(PS)、およびIO発行回数(IF1(i))とIO待ち時間(IT)から算出する。このように、通信プロセッサモジュール(12)は図23の算出式を用いて、受信側プロセッサモジュール(14~19)の処理時間(RT(j), RF(j))を計算する。

【0073】そして、通信プロセッサモジュール(12)は、すべての受信側プロセッサモジュール(14~19)の処理時間が一定値(L1)以上かどうかチェックする(処理403)。そして、これが一定値以上(L1)ならば、このプロセッサモジュールは輻輳状態(151)と判定し、このプロセッサモジュールへの電文送信を禁止する(処理404)。

【0074】さらに、すべてのファイルプロセッサモジュール(17~19)が輻輳状態(151)かどうか判定する(処理405)。すべてのファイルプロセッサモジュール(17~19)が輻輳状態(151)ならば、トランザクションプロセッサモジュール(11~13)に対して電文送信禁止を通知する(処理406)。ここで、この通知する電文を図24(2)に示す。ここでは通信プロセッサモジュール(12)に送信する場合を示す。送信元アドレス(80)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(81)はプロセッサモジュール(11)であるため(00)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16を、システム稼働情報2(84)は輻輳発生であるため(01)16をそれぞれ設定する。

【0075】また、受信側プロセッサモジュール(14~19)が輻輳状態(151)であるかを判定し(処理407)、輻輳状態(151)のプロセッサモジュールの処理時間が一定値(L2)以下になっている(処理408)と、下位プロセッサモジュールへの電文送信を再開し、上位のプロセッサモジュールに対して電文送信再開を通知する(処理409)。

【0076】そして、ケース1~3の場合には、コンソール(8)に通知する(処理410)。ここで、ケース1の通知する電文を図24(3)に示す。送信元アドレス(80)はプロセッサモジュール(12)であるため(01)16を、送信先アドレス(81)はコンソール

(8)であるため(30)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16を、システム稼働情報2(84)は1つのプロセッサモジュールで輻輳発生であるため(01)16をそれぞれ設定する。

【0077】ここで、ケース2の通知する電文を図24(4)に示す。送信元アドレス(80)、送信先アドレス(81)、制御用電文種別(82)、システム稼働情報1(83)はケース1と同一である。システム稼働情報2(84)は、すべてのファイルプロセッサモジュール(17~19)で輻輳発生であるため、(02)16を設定する。

【0078】ここで、ケース3の通知する電文を図24(5)に示す。送信元アドレス(80)、送信先アドレス(81)、制御用電文種別(82)、システム稼働情報1(83)はケース1と同一であり、システム稼働情報2(84)は、ファイルプロセッサモジュール(17~19)は輻輳解除であるため、(10)16を設定する。このようにして、受信側プロセッサモジュール(14~19)は、通信プロセッサモジュール(12)あるいはコンソール(8)に輻輳発生および輻輳解除を通知できる。

【0079】次に、図25から図27より、障害時および障害回復時の処理手順について説明する。図25は障害時の処理手順を示す図である。図27に障害時の制御用電文を示す。

【0080】ファイルプロセッサモジュール(19)は、一定間隔毎に常にファイルプロセッサモジュール(17)にaliveメッセージを送信している(処理500)。ファイルプロセッサモジュール(17)は、aliveメッセージの受信によりファイルプロセッサモジュール(19)が正常に動作していると認識している。ファイルプロセッサモジュール(19)で障害が発生すると(処理501)、ファイルプロセッサモジュール(17)はaliveメッセージの途絶により、ファイルプロセッサモジュール(19)の障害を検出する(処理502)。

【0081】ここで、正常動作を通知する電文を図27(1)に示す。ファイルプロセッサモジュール(19)がファイルプロセッサモジュール(17)にリセットメッセージを送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(19)であるため(22)16を、送信先アドレス(81)はファイルプロセッサモジュール(17)であるため(20)16を、制御用電文種別(82)はaliveメッセージであるため(01)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため(FF)16を、それぞれ設定する。

【0082】更に、ファイルプロセッサモジュール(1

7)は障害の発生したファイルプロセッサモジュール(19)にリセットを要求する(処理503)。そして、ファイルプロセッサモジュール(19)はモニタ(19-20)がIO制御装置(19-10)をリセットする処理を実行する(処理504)。

【0083】ここで、リセット要求を通知する電文を図27(2)に示す。ファイルプロセッサモジュール(18)がファイルプロセッサモジュール(19)にリセットメッセージを送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)はファイルプロセッサモジュール(19)であるため(22)16を、制御用電文種別(82)はリセットメッセージであるため(10)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため(FF)16をそれぞれ設定する。

【0084】更に、ファイルプロセッサモジュール(17)は障害発生をコンソール(8)に通知し(処理505)、通信プロセッサモジュール(12)に通知する(処理506)。そして、通信プロセッサモジュール(12)はこれを受信する(処理507)。

【0085】ここで、コンソールに障害発生を通知する電文を図27(3)に示す。プロセッサモジュール(17)がコンソール(8)に障害発生を送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)はコンソール(8)であるため(30)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16、システム稼働情報2(84)は障害発生通知であるため(03)16をそれぞれ設定する。

【0086】次に、通信プロセッサモジュールに障害発生を通知する電文を図27(4)に示す。ファイルプロセッサモジュール(17)が通信プロセッサモジュール(12)に障害発生を送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)は通信プロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はプロセッサモジュール情報であるため(10)16、システム稼働情報2(84)は障害発生通知であるため(03)16をそれぞれ設定する。

【0087】図26は障害回復時の処理手順を示す図である。図27(5)は障害回復時の制御用電文を示す図である。障害の発生したファイルプロセッサモジュール(19)は障害から回復したことを通信プロセッサモジュール(12)に通知する(処理550)。そして、通信プロセッサモジュール(12)は障害回復の通知を

受信する(処理 551)。

【0088】図27(5)は通信プロセッサモジュールに障害回復を通知する電文である。ファイルプロセッサモジュール(19)が通信プロセッサモジュール(12)に障害回復を送信する場合を示す。送信元アドレス(80)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(81)は通信プロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16、システム稼働情報2(84)は障害回復通知であるため(11)16をそれぞれ設定する。

【0089】最後に、障害回復の通知をコンソール(8)に表示する(処理 552)。次に、コンソールに障害回復を通知する電文を図27(6)に示す。通信プロセッサモジュール(12)がコンソール(8)に障害回復を送信する場合を示す。送信元アドレス(80)は通信プロセッサモジュール(12)であるため(01)16を、送信先アドレス(81)はコンソール(8)であるため(30)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16、システム稼働情報2(84)は障害回復通知であるため(11)16をそれぞれ設定する。

【0090】このようにして、プロセッサモジュール(11~19)間あるいはプロセッサモジュール(11~19)とコンソール(8)間で制御用通信バス(1)を介して通信を行うことが可能となる。

【0091】まず、図28と図31を用いて、3つのプロセッサモジュール(11~19)間での電文処理手順について説明する。図28は電文処理手順を示す図である。図29はプロセス間通信の電文を示す図である。通信プロセッサモジュール(12)は、端末(6)から電文を受信すると(処理 600)、各トランザクションプロセッサモジュール(14~16)の処理時間を算出し、このうち最も処理時間の短いプロセッサモジュールを選ぶ(処理 601)。図28では、トランザクションプロセッサモジュール(15)とファイルプロセッサモジュール(19)とが選択されたとする。通信プロセッサモジュール(12)はトランザクションプロセッサモジュール(15)に電文を送信する。

【0092】図29(1)は、通信プロセッサモジュール(12)がトランザクションプロセッサモジュール(15)に送信する電文を示す。送信元アドレス(90)は通信プロセッサモジュール(12)であるため(01)16を、送信先アドレス(91)はトランザクションプロセッサモジュール(15)であるため(11)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はファイル

プロセッサモジュール(19)であるため(22)16、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0093】トランザクションプロセッサモジュール(15)はファイルプロセッサモジュール(19)にディスク(2-2)からの読み出しを要求する(処理 605)。図29(2)に読み出し要求の電文を示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はファイルプロセッサモジュール(19)であるため(22)16を、電文本体(92)は(020301)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0094】ファイルプロセッサモジュール(19)は読み出し要求を受信すると(処理606)、ディスク(2-2)の読み出し処理を実行する(処理 607)。そして、読み出し結果をトランザクションプロセッサモジュール(15)に送信する(処理 608)。図29(3)に読み出し結果送信の電文を示す。送信元アドレス(90)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(91)はプロセッサモジュール(15)であるため(11)16を、電文本体(92)は(040203)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16を、それぞれ設定する。

【0095】さらに、トランザクションプロセッサモジュール(15)はファイルプロセッサモジュール(19)にディスク(2-2)への書き込みを要求する(処理 609)。図29(4)に書き込み要求の電文を示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はプロセッサモジュール(19)であるため(22)16を、電文本体(92)は(050403)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16、電文の識別子(94)は(0001)16を、それぞれ設定する。

【0096】ファイルプロセッサモジュール(19)は書き込み要求を受信すると(処理610)、ディスク(2-2)の書き込み処理を実行し(処理 611)、書き込み完了を送信する(処理 612)。図29(5)に書き込み完了の電文を示す。送信元アドレス(90)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(91)はプロセッサモジュール(15)であるため(11)16を、電文本体(92)は(070203)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16を、それぞれ設定する。

【0097】トランザクションプロセッサモジュール(15)は通信プロセッサモジュール(12)に処理結果を送信する(処理 613)。図29(6)に処理結果送信の電文を示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はプロセッサモジュール(12)であるため(01)16を、電文本体(92)は(010503)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16を、それぞれ設定する。

【0098】最後に、通信プロセッサモジュール(12)は、端末(6)に処理結果を送信する(処理 614)。このように、プロセッサモジュール(14~16、17~19)間の負荷分散が可能となる。

【0099】図30はトランザクションプロセッサモジュールで障害が発生した場合の処理を示す図である。図31はファイルプロセッサモジュールで障害が発生した場合の処理を示す図である。図32はプロセス間通信の電文を示す図である。まず、図30と図32を用いて、トランザクションプロセッサモジュール(15)で障害が発生した場合について説明する。

【0100】トランザクションプロセッサモジュール(15)で障害が発生とすると(処理650)、トランザクションプロセッサモジュール(16)はaliveメッセージの途絶により、障害を検出する(処理 651)。そして、トランザクションプロセッサモジュール(15)に対し、リセットを要求し(処理 652)、トランザクションプロセッサモジュール(15)はリセット処理を実行する(処理653)。

【0101】リセット要求の制御用電文を図32(1)に示す。送信元アドレス(80)はプロセッサモジュール(16)であるため(12)16を、送信先アドレス(81)はプロセッサモジュール(15)であるため(11)16を、制御用電文種別(82)はリセットメッセージであるため(10)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため、(FF)16を格納する。そして、トランザクションプロセッサモジュール(16)は、通信プロセッサモジュール(12)に、障害発生を通知する(処理 654)。

【0102】障害通知の制御用電文を図32(2)に示す。送信元アドレス(80)はプロセッサモジュール(16)であるため(12)16を、送信先アドレス(81)はプロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16を、システム稼働情報2(84)は障害通知であるため(03)16を、それぞれ格納する。

【0103】通信プロセッサモジュール(12)はトラ

ンザクションプロセッサモジュール(15)の処理時間を算出し、別のトランザクションプロセッサモジュール(16)に電文を送信する(処理 655)。そして、トランザクションプロセッサモジュール(16)は、電文を実行しているファイルプロセッサモジュールに通知する(処理 656)。そして、トランザクションプロセッサモジュール(16)は、ファイルプロセッサモジュール(19)に電文を送信する(処理 657)。

【0104】プロセス間通信の電文を図32(3)に示す。送信元アドレス(90)はプロセッサモジュール(12)であるため(01)16を、送信先アドレス(91)はプロセッサモジュール(15)であるため(11)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はプロセッサモジュール(19)であるため(22)16、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0105】トランザクションプロセッサモジュール(15)は電文を受信すると(処理658)、ファイルプロセッサモジュール(19)に読み出し要求を送信する(処理 659)。

【0106】一方、ファイルプロセッサモジュール(19)は、電文を受信すると、チェックポイント情報に基づいて、回復処理を行う。書き込み処理が完了していれば、処理結果をトランザクションプロセッサモジュールに戻し、書き込み処理が完了していなければ、最初から電文処理を再実行する。

【0107】次に、図31と図32を用いて、ファイルプロセッサモジュール(19)で障害が発生した場合について説明する。ファイルプロセッサモジュール(19)で障害が発生とすると(処理 670)、ファイルプロセッサモジュール(17)はaliveメッセージの途絶により、障害を検出する(処理 671)。そして、ファイルプロセッサモジュール(19)に対し、リセットを要求し(処理 672)、ファイルプロセッサモジュール(19)はリセット処理を実行する(処理 673)。

【0108】リセット要求の制御用電文を図32(4)に示す。送信元アドレス(80)はプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)はプロセッサモジュール(19)であるため(22)16を、制御用電文種別(82)はリセットメッセージであるため(10)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため、(FF)16を格納する。

【0109】そして、ファイルプロセッサモジュール(17)は、通信プロセッサモジュール(12)に、障害発生を通知する(処理 674)。

【0110】障害通知の制御用電文を図32(5)に示す。送信元アドレス(80)はプロセッサモジュール

(17)であるため(20)16を、送信先アドレス(81)はプロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16を、システム稼働情報2(84)は障害通知であるため(03)16を、それぞれ格納する。

【0111】通信プロセッサモジュール(12)は、ファイルプロセッサモジュール(17)を介してファイルプロセッサモジュール(19)の障害発生を受信する(処理675)。通信プロセッサモジュール(12)はトランザクションモジュール(14~16)の処理時間を算出し、処理時間の最も短いプロセッサモジュール(15)を選択する(処理676)。そして、通信プロセッサモジュール(12)は、トランザクションプロセッサモジュール(15)に送信済みの電文を再送する(処理677)。

【0112】プロセス間通信の電文を図32(6)に示す。送信元アドレス(90)はプロセッサモジュール(12)であるため(01)16を、送信先アドレス(91)はプロセッサモジュール(15)であるため(11)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はプロセッサモジュール(19)であるため(22)16、電文の識別子(94)は(0001)16をそれぞれ設定する。トランザクションプロセッサモジュール(15)は電文を受信すると(処理678)、ファイルプロセッサモジュール(19)に読み出し要求を送信する(処理679)。

【0113】一方、ファイルプロセッサモジュール(19)は、電文を受信すると、ジャーナルに基づいて、回復処理を行う。書き込み処理が完了していれば、処理結果をトランザクションプロセッサモジュール(15)に戻し、書き込み処理が完了していなければ、最初から電文処理を再実行する。

【0114】通信プロセッサモジュール(12)で障害が発生すると、一定時間経過しても、応答を戻らないことにより、端末(6)が通信プロセッサモジュール(12)の障害を検出し、別の通信プロセッサモジュール(13)に通知する。この結果、プロセッサモジュール(15、19)で障害が発生すると、電文の再送処理が必須となり、回復時間が長くなるという問題が残る。

【0115】最後に、予備のプロセッサモジュール(16、17、13)を考慮し、電文を予備のプロセッサモジュール(16、17、13)にも送信し、障害時の回復時間を短縮させる手順を示す。

【0116】図33は予備のプロセッサモジュールを考慮した場合の処理を示す図である。図34は予備のプロセッサモジュールを考慮した場合のプロセス間通信の電文を示す図である。

【0117】端末(6)は、業務用LAN(4)経由の場合には通信プロセッサモジュール(12、13)に電文を送信する。回線切替装置(5)経由の場合には、回線切替装置(5)は、2つの通信プロセッサモジュール(12、13)が電文を受信できるように制御を行い、端末(6)からの電文を通信プロセッサモジュール(12、13)が受信する(処理700、701)。本実施例では、プロセッサモジュール(13)を予備の通信プロセッサモジュールとする。実際に通信処理を実行する通信プロセッサモジュール(12)は、下位のプロセッサモジュール(14~19)の処理時間を計算し、処理時間の最も短いプロセッサモジュールと、予備のプロセッサモジュールを決める(処理702)。そして、処理時間が最小となるプロセッサモジュールはトランザクションプロセッサモジュール(15)とファイルプロセッサモジュール(19)とする。更に、それぞれの予備のプロセッサモジュールをトランザクションプロセッサモジュール(16)とファイルプロセッサモジュール(17)とする。

【0118】通信プロセッサモジュール(12)は、電文をトランザクションプロセッサモジュール(15、16)に送信する(処理703)。このプロセス間通信の電文を図34(1)に示す。送信元アドレス(90)はプロセッサモジュール(12)であるため(01)16を、送信先アドレス(91)はプロセッサモジュール(15、16)であるため(1112)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はプロセッサモジュール(19、17)であるため(22、20)16を、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0119】トランザクションプロセッサモジュール(15)は電文を受信すると(処理704)、読み出し要求をファイルプロセッサモジュール(19、17)に送信する(処理706)。このプロセス間通信の電文を図34(2)に示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はプロセッサモジュール(19、17)であるため(2220)16を、電文本体(92)は(020301)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16を、それぞれ設定する。一方、トランザクションプロセッサモジュール(16)は電文を受信するだけである(処理705)。

【0120】ファイルプロセッサモジュール(19)は、読み出し要求を受信すると(処理707)、ディスク(2-2)からの読み出し処理を実行する(処理709)。ファイルプロセッサモジュール(17)は電文を受信するだけである(処理708)。そして、ファイルプロセッサモジュール(19)は、トランザクションプロセッサモジュール(15、16)に読み出し結果を

送信する(処理 710)。このプロセス間通信の電文を図34(3)に示す。送信元アドレス(90)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(91)はプロセッサモジュール(15、16)であるため(1112)16を、電文本体(92)は(040203)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0121】トランザクションプロセッサモジュール(15)は、読み出し結果を受信すると(処理 711)、更に処理を継続させ、書き込み要求をファイルプロセッサモジュール(19、17)に送信する(処理 713)。このプロセス間通信の電文を図34(4)に示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はプロセッサモジュール(19、17)であるため(2220)16を、電文本体(92)は(050403)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16をそれぞれ設定する。一方、トランザクションプロセッサモジュール(16)は読み出し結果を受信するだけである(処理 712)。

【0122】ファイルプロセッサモジュール(19)は書き込み処理を受信すると(処理714)、ディスク(2-2)への書き込み処理を実行する(処理 716)。そして、ファイルプロセッサモジュール(19)は、書き込み完了をトランザクションプロセッサモジュール(15、16)に送信する(処理 717)。このプロセス間通信の電文を図34(5)に示す。送信元アドレス(90)はプロセッサモジュール(19)であるため(22)16を、送信先アドレス(91)はプロセッサモジュール(15、16)であるため(1112)16を、電文本体(92)は(070203)16を、下位のプロセッサモジュールのアドレス(93)はないため(FF)16を、電文の識別子(94)は(0001)16をそれぞれ設定する。一方、ファイルプロセッサモジュール(17)は書き込み要求を受信するだけである(処理 715)。

【0123】トランザクションプロセッサモジュール(15)は処理結果を受信する(処理719)と、通信プロセッサモジュール(12、13)に処理結果を送信する(処理 721)。このプロセス間通信の電文を図34(6)に示す。送信元アドレス(90)はプロセッサモジュール(15)であるため(11)16を、送信先アドレス(91)はプロセッサモジュール(11、12)であるため(0102)16を、電文本体(92)は(010503)16を、下位のプロセッサモジュールのアドレス(93)はないため、(FF)16を、電文の識別子(94)は(0001)16をそれぞれ設定する。トランザクションプロセッサモジュール(16)は処理結果

を受信するだけである(処理 720)。

【0124】最後に、通信プロセッサモジュール(12)は、処理結果を受信すると(処理722)、端末(6)に送信する(処理 724)。通信プロセッサモジュール(12)は処理結果を受信するだけである(処理 723)。

【0125】次に、各プロセッサモジュール(15、19)で障害が発生した場合について、図35から図37を用いて説明する。図35は、トランザクションプロセッサモジュールで障害が発生した場合の処理を示す図である。図37はプロセス間通信の電文を示す図である。

【0126】まず、トランザクションプロセッサモジュール(15)で障害が発生した場合について、図35と図37を用いて説明する。図35はトランザクションプロセッサモジュールで障害が発生した場合の処理を示す図である。

【0127】トランザクションプロセッサモジュール(15、16)は、通信プロセッサモジュール(12)から同時に電文を受信している(処理 703~705)。トランザクションプロセッサモジュール(15)で障害が発生すると(処理 730)、トランザクションプロセッサモジュール(16)は、aliveメッセージの途絶により、トランザクションプロセッサモジュール(15)の障害を検出する(処理 731)。

【0128】更に、トランザクションプロセッサモジュール(16)は、障害の発生したトランザクションプロセッサモジュール(15)にリセットを要求する(処理 732)。そして、トランザクションプロセッサモジュール(15)は、モニタ(15-20)がIO制御装置(15-10)をリセットする処理を実行する(処理 733)。

【0129】ここで、リセット要求を通知する電文を図37(1)に示す。トランザクションプロセッサモジュール(16)がトランザクションプロセッサモジュール(15)にリセットメッセージを送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(16)であるため(12)16を、送信先アドレス(81)はファイルプロセッサモジュール(15)であるため(11)16を、制御用電文種別(82)はリセットメッセージであるため(10)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため(FF)16を、それぞれ設定する。

【0130】更に、トランザクションプロセッサモジュール(16)は、障害発生を通信プロセッサモジュール(12)に通知する(処理 734)。そして、通信プロセッサモジュール(12)は障害発生の通知を受信する(処理 735)。障害発生を通知する電文を図37(2)に示す。トランザクションプロセッサモジュール(16)が通信プロセッサモジュール(12)に障害発生を送信する場合を示す。送信元アドレス(80)はト

ランザクションプロセッサモジュール(16)であるため(12)16を、送信先アドレス(81)は通信プロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16、システム稼働情報2(84)は障害発生通知であるため(03)16をそれぞれ設定する。

【0131】そして、トランザクションプロセッサモジュール(16)は、再度読み出し要求をファイルプロセッサモジュール(19、17)に送信する(処理 736)。読み出し要求を通知する電文を図37(3)に示す。送信元アドレス(90)はトランザクションプロセッサモジュール(16)であるため(12)16を、送信先アドレス(91)はトランザクションプロセッサモジュール(19、17)であるため(2220)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はないため、(FF)16、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0132】次に、ファイルプロセッサモジュール(15)で障害が発生した場合について、図36と図37を用いて説明する。図36はファイルプロセッサモジュールで障害が発生した場合の処理を示す図である。

【0133】ファイルプロセッサモジュール(19、17)は、トランザクションプロセッサモジュール(15)から同時に電文を受信している(処理 706~708)。ファイルプロセッサモジュール(19)で障害が発生すると(処理 760)、ファイルプロセッサモジュール(17)はaliveメッセージの途絶により、ファイルプロセッサモジュール(19)の障害を検出する(処理 761)。

【0134】更に、ファイルプロセッサモジュール(17)は、障害の発生したファイルプロセッサモジュール(19)にリセットを要求する(処理 762)。そして、ファイルプロセッサモジュール(19)はモニタ(19-20)がIO制御装置(19-10)をリセットする処理を実行する(処理 763)。

【0135】ここで、リセット要求を通知する電文を図37(4)に示す。ファイルプロセッサモジュール(17)がファイルプロセッサモジュール(19)にリセットメッセージを送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)はファイルプロセッサモジュール(19)であるため(21)16を、制御用電文種別(82)はリセットメッセージであるため(10)16を、システム稼働情報1(83)とシステム稼働情報2(84)は使用しないため(FF)16を、それぞれ設定する。

【0136】更に、ファイルプロセッサモジュール(1

7)は障害発生を通信プロセッサモジュール(12)に通知する(処理 764)。そして、通信プロセッサモジュール(12)はこれを受信する(処理 765)。障害発生を通知する電文を図37(5)に示す。ファイルプロセッサモジュール(17)が通信プロセッサモジュール(12)に障害発生を送信する場合を示す。送信元アドレス(80)はファイルプロセッサモジュール(17)であるため(20)16を、送信先アドレス(81)は通信プロセッサモジュール(12)であるため(01)16を、制御用電文種別(82)はシステム稼働情報であるため(02)16を、システム稼働情報1(83)はイベント情報であるため(10)16、システム稼働情報2(84)は障害発生通知であるため(03)16をそれぞれ設定する。

【0137】そして、ファイルプロセッサモジュール(17)は、再度読み出し処理を実行する(処理 766)。そして、ファイルプロセッサモジュール(17)は、読み出し結果をトランザクションプロセッサモジュール(15、16)に送信する(処理 767)。読み出し結果の送信を通知する電文を図37(6)に示す。送信元アドレス(90)はファイルプロセッサモジュール(17)であるため(12)16を、送信先アドレス(91)はトランザクションプロセッサモジュール(15、16)であるため(1112)16を、電文本体(92)は(010203)16を、下位のプロセッサモジュールのアドレス(93)はないため、(FF)16、電文の識別子(94)は(0001)16をそれぞれ設定する。

【0138】通信プロセッサモジュール(12)で障害が発生すると、端末(6)は両方の通信プロセッサモジュール(12、13)に電文を送信することにより、端末(6)への電文の再送要求をなくし、回復時間を短縮できる。本実施例では、電文処理を3つに分割し、3階層の場合について説明した。同様に、電文処理をn個に分割したn階層のクラスタ型計算機システムにも本方式を適用できる。

【0139】通信プロセッサモジュール(11~13)(最上位のプロセッサモジュール)が、受信側プロセッサモジュールの入力待ち行列に格納されている電文情報から、受信した電文の処理時間を計算し、これが最小となるプロセッサモジュールを見つけ、これらのプロセッサモジュールで電文処理を実行することにより、各プロセスのプロセッサモジュールの負荷を均一にし、システム全体の処理能力を向上させることが可能となる。

【0140】更に、処理時間が2番目に短いプロセッサモジュールを予備のプロセッサモジュールとし、予備のプロセッサモジュールにも電文を送信することにより、回復時間を短縮させることが可能となる。

【0141】図38は本発明によるシステム構成図(2)である。ここでは、本発明によるシステム構成は、図1のシステム構成と同様に9個のプロセッサモジ

ジュール(11~19)を前提とする。すべてのプロセッサモジュール(11~19)は、制御用通信バス(1)およびプロセス間通信バス(3)により接続される。図38では、プロセッサモジュールに2つの異なるプロセスを搭載させ、相互にバックアップさせるものである。

【0142】プロセッサモジュール(11~13)は、図5で説明した通信プロセス(30)を搭載する。プロセッサモジュール(11~13)は、共有ディスク(2-0)を共有し、障害発生時に、処理を引き継ぐために必要なチェックポイント情報を格納する。

【0143】プロセッサモジュール(14~16)のそれぞれは、トランザクションプロセス(31)とファイルプロセス(32)を搭載する。トランザクションプロセス(31)は現用プロセスとして稼働し、また、ファイルプロセス(32-1)は待機プロセスとして稼働される。

【0144】プロセッサモジュール(17~19)のそれぞれは、トランザクションプロセス(31)とファイルプロセス(32)を搭載する。ファイルプロセス(32)は現用プロセスとして稼働し、また、トランザクションプロセス(31-1)は待機プロセスとして稼働される。

【0145】プロセッサモジュール(14~19)の障害を検出した際に、プロセス(31, 32)の個数に応じて、システムの処理能力が最高になるように、プロセス毎のプロセッサモジュールを決め、プロセス間での処理を継続することにより、システム全体の処理能力を向上させる。

【0146】トランザクションプロセッサモジュール(15)で障害が発生した場合を想定する。ここで、トランザクションプロセッサモジュール(14~16)とファイルプロセッサモジュール(17~19)はそれぞれ2台と3台で稼働すべきか、あるいは3台と2台で稼働すべきかを判定する。そして、システム処理能力の大きな方を選ぶ。

【0147】そして、前者の場合には、障害の発生したトランザクションプロセッサモジュールを閉塞する。後者の場合には、1つのファイルプロセッサモジュール(19)がトランザクションプロセス(31)を引き継ぐ。このように、各プロセッサモジュール(11~19)に複数のプロセス(30~32)を搭載させる構成も可能となる。

【0148】これにより、1つのトランザクションプロセス(31)で障害が発生しても、プロセス単位のホットスタンバイ機能により、3つのトランザクションプロセス(31)と2つのファイルプロセス(32)といったシステムを柔軟に再構築することが可能となり、障害が発生しても、システム処理能力の低下を最小限にすることが可能となる。

【0149】

【発明の効果】本発明では、複数のプロセッサモジュールからなるシステムにおいて、受信側プロセッサモジュールが通信プロセッサモジュールにシステム稼働情報を送信し、通信プロセッサモジュールが処理時間の最小となる受信側プロセッサモジュールに電文を送信することにより、また、予備のプロセッサモジュールにも電文を送信することにより、プロセッサモジュールの負荷を均一にし、システム全体の処理能力が向上でき、更に、プロセッサモジュール障害時の回復時間を短縮できる。

【図面の簡単な説明】

【図1】本発明によるシステム構成図である。

【図2】本発明の処理概要を示す図である。

【図3】プロセッサモジュールの構成図である。

【図4】電文処理の概要を示す図である。

【図5】ソフトウェアの構成と電文処理の分割を示す図である。

【図6】IO制御装置のシステム構成図である。

【図7】プロセッサモジュールの状態遷移図である。

【図8】プロセッサモジュールの状態コードを示す図である。

【図9】制御用通信バスにおける電文フォーマットを示す図である。

【図10】制御用電文種別を示す図である。

【図11】システム稼働情報1とシステム稼働情報2を示す図である。

【図12】電文のタイプとその個数を示す図である。

【図13】電文のタイプとそのコードを示す図である。

【図14】プロセッサモジュールとコンソールのアドレスを示す図である。

【図15】プロセス間通信の電文フォーマットを示す図である。

【図16】トランザクションプロセッサモジュールの入力待ち行列に格納されている電文を示す図である。

【図17】ファイルプロセッサモジュールの入力待ち行列に格納されている電文を示す図である。

【図18】トランザクションプロセッサモジュールの電文毎の処理ステップ数とファイルプロセッサモジュールとの通信回数を示す図である。

【図19】ファイルプロセッサモジュールの電文毎の処理ステップ数とIO発行回数を示す図である。

【図20】トランザクションプロセッサモジュールの処理能力と状態を示す図である。

【図21】ファイルプロセッサモジュールの処理能力と状態を示す図である。

【図22】システム稼働情報による処理手順を示す図である。

【図23】通信プロセッサモジュールの処理時間の算出式である。

【図24】制御用電文の一例を示す図である。

【図25】障害時の処理手順を示す図である。

【図26】障害回復時の処理手順を示す図である。
 【図27】障害時の制御用電文を示す図である。
 【図28】電文処理手順を示す図である。
 【図29】プロセス間通信電文を示す図である。
 【図30】トランザクションプロセッサモジュールで障害が発生した場合の処理を示す図である。
 【図31】ファイルプロセッサモジュールで障害が発生した場合の処理を示す図である。
 【図32】プロセス間通信の電文を示す図である。
 【図33】予備のプロセッサモジュールを考慮した場合の処理を示す図である。
 【図34】予備のプロセッサモジュールを考慮した場合のプロセス間通信の電文を示す図である。

【図35】トランザクションプロセッサモジュールで障害が発生した場合の処理を示す図である。

【図36】ファイルプロセッサモジュールで障害が発生した場合の処理を示す図である。

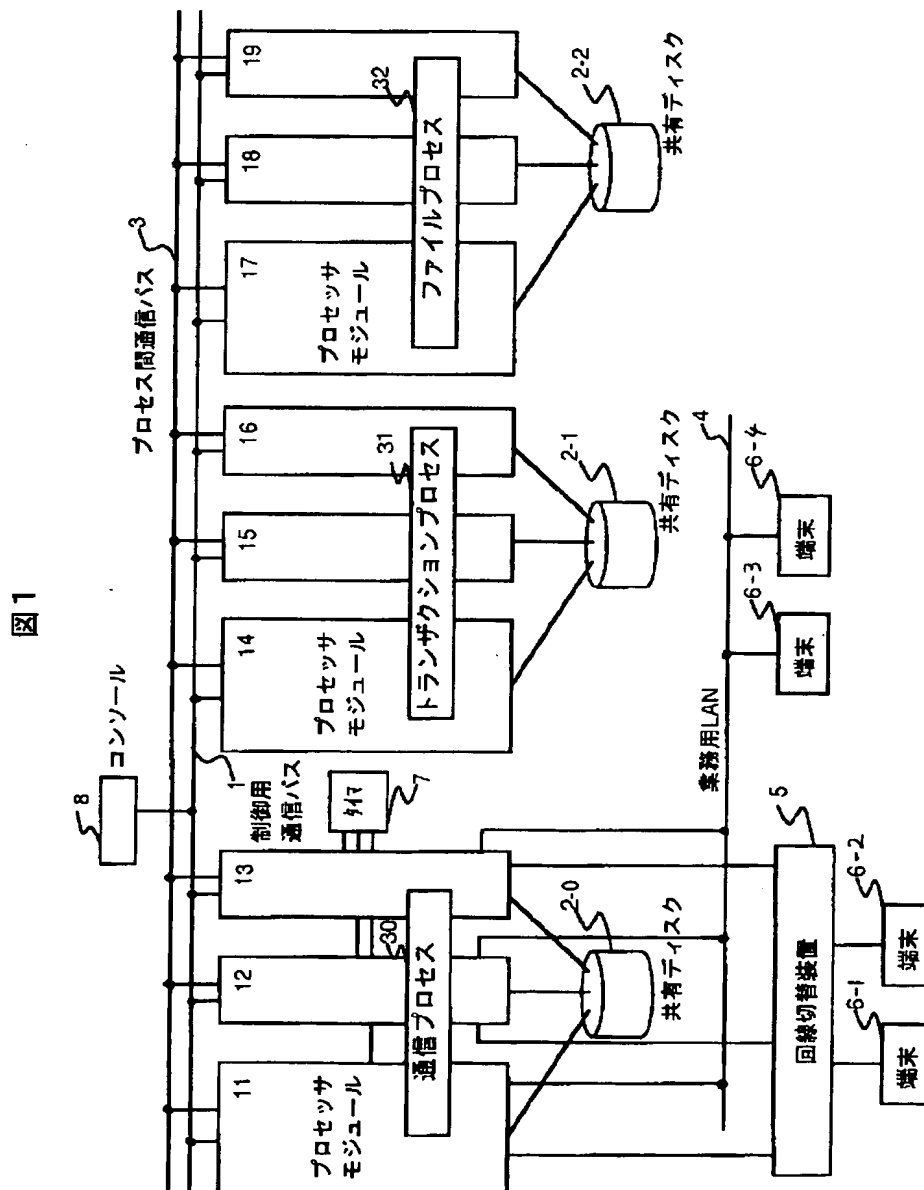
【図37】プロセス間通信の電文を示す図である。

【図38】本発明によるシステム構成図(2)である。

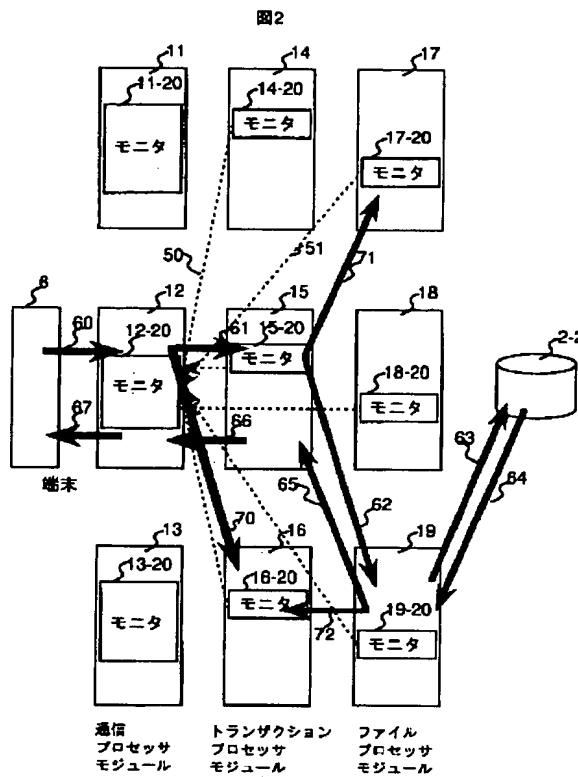
【符号の説明】

1…制御用通信バス、2-0～2-2…共有ディスク、3…プロセス間通信バス、4…業務用LAN、5…回線切替装置、6…端末、7…タイマ、11～19…プロセッサモジュール、30…通信プロセス、31…トランザクションプロセス、32…ファイルプロセス

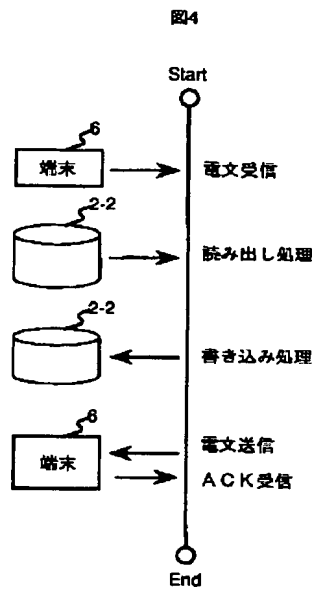
【図1】



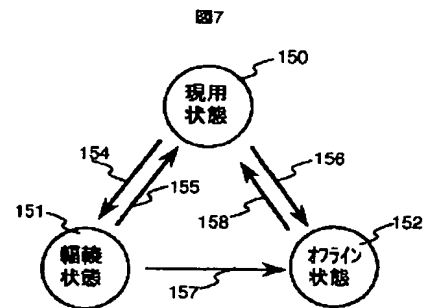
【図2】



【図4】



【図7】

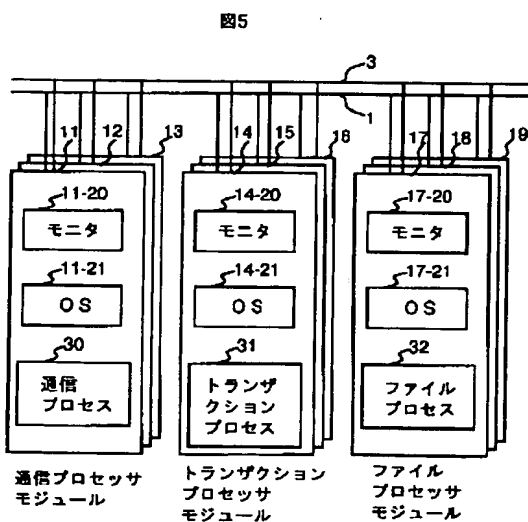


【図13】

図13

電文のタイプ	コード
電文1	(0 1)16
電文2	(0 2)16
電文3	(0 3)16
電文4	(0 4)16
電文5	(0 5)16

【図5】



【図8】

図8

プロセッサモジュールの状態	コード
現用状態	(0 1)16
輻輳状態	(0 2)16
ワライン状態	(0 3)16

【図19】

図19

電文種別	実行ステップ数	通信回数	IO回数
	SF1(i)	CF1(i)	IF1(i)
電文1	100000	1	4
電文2	150000	1	5
電文3	130000	1	3
電文4	150000	1	4
電文5	170000	1	7

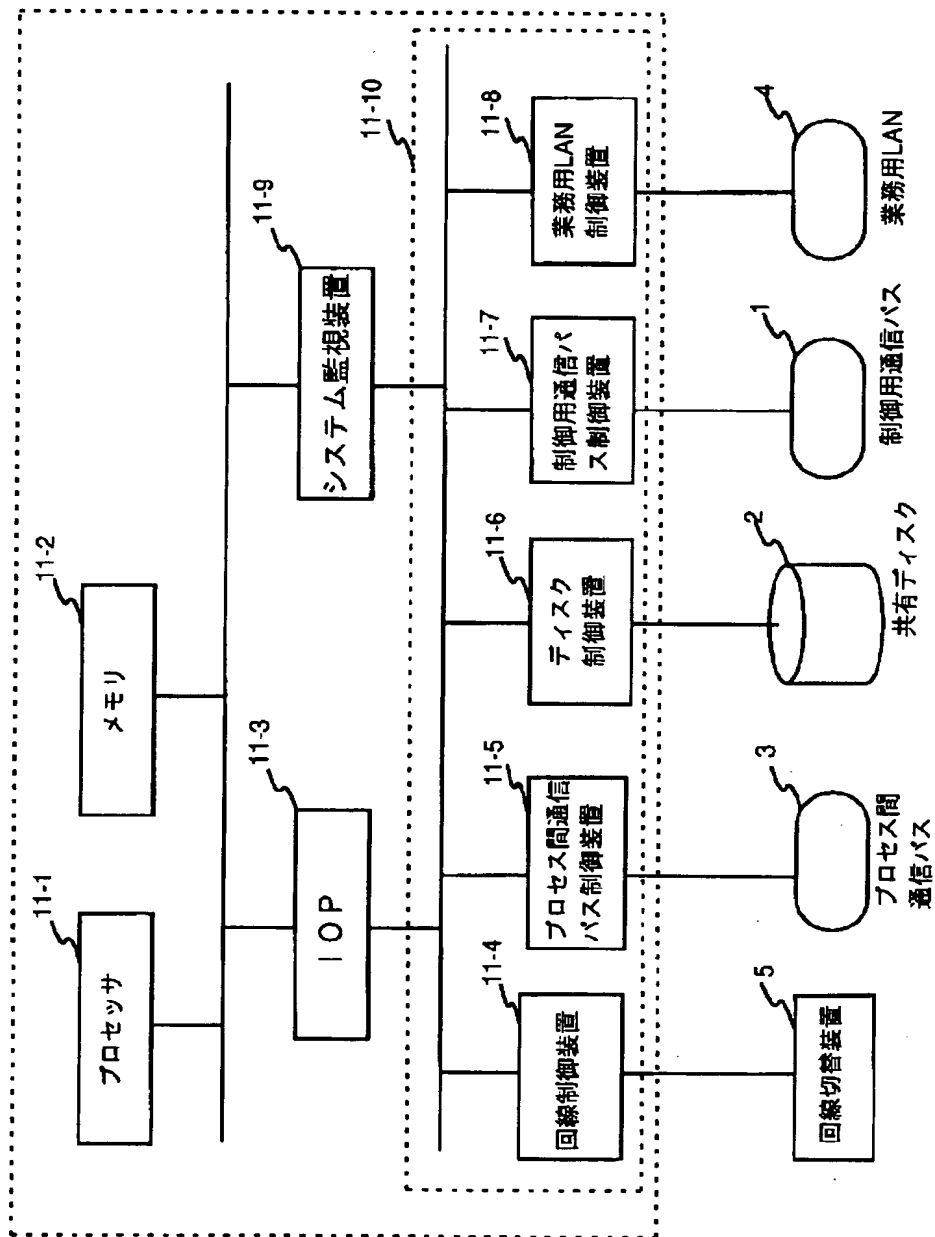
【図9】

図9

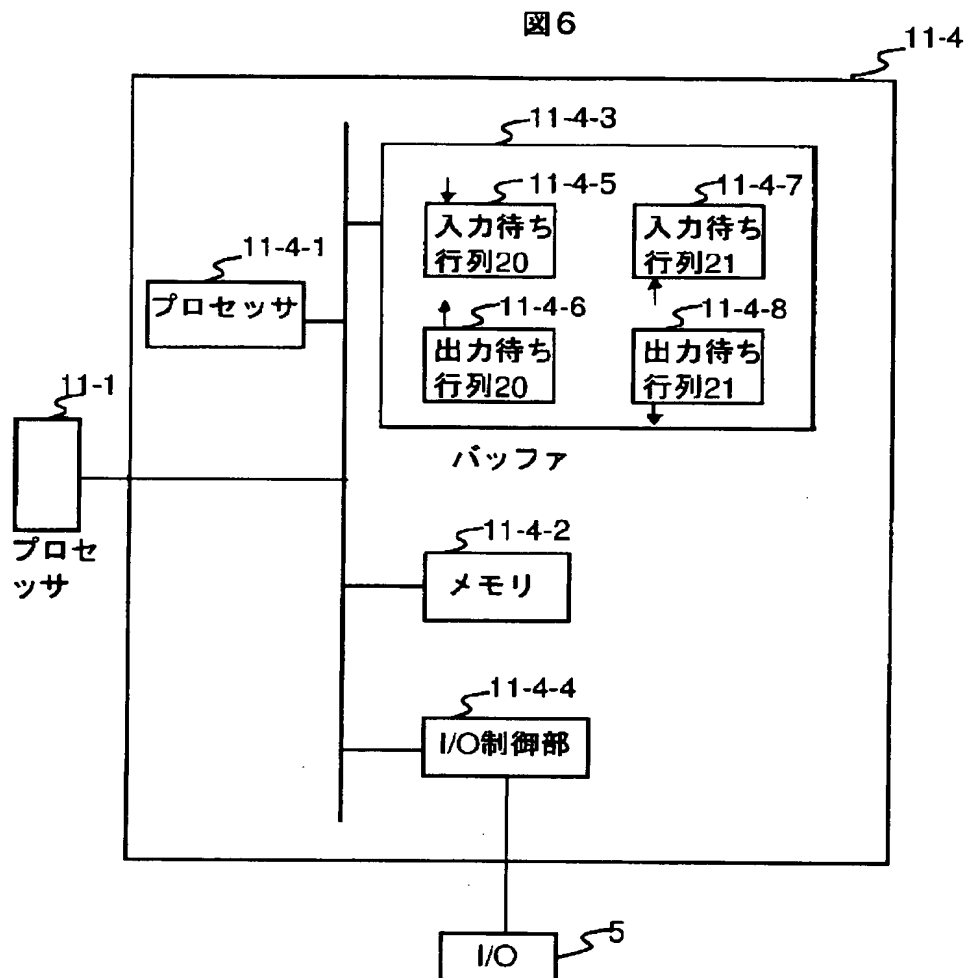
送信元アドレス	送信先アドレス	制御用電文種別	システム稼働情報1	システム稼働情報2
80	81	82	83	84

【図3】

図3



【図6】



【図10】

図10

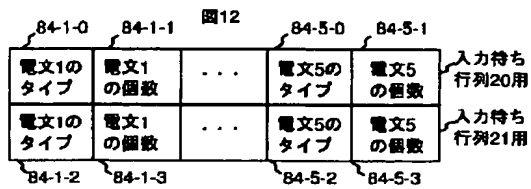
制御用電文	コード
リセットメッセージ	(10)16
aliveメッセージ	(01)16
システム稼働情報	(02)16

【図11】

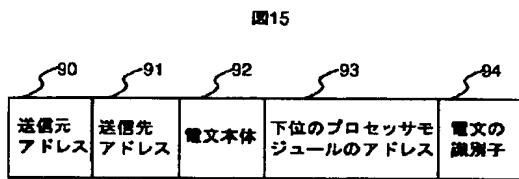
図11

システム稼働情報1	システム稼働情報2	意味
(01)16	図12参照	電文のタイプと個数
(10)16	(01)16	1つのプロセッサモジュールが稼働発生
	(02)16	同一プロセスの全プロセッサモジュールで稼働発生
	(03)16	障害発生時の通知
	(10)16	稼働解除通知
	(11)16	障害回復通知

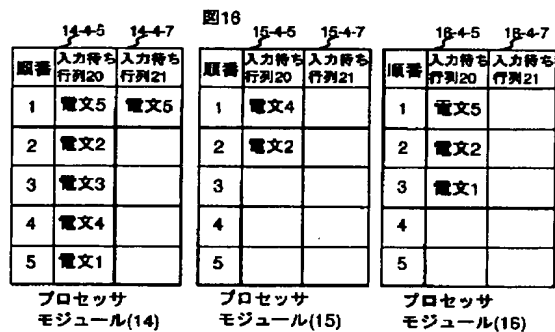
【図12】



【図15】



【図16】

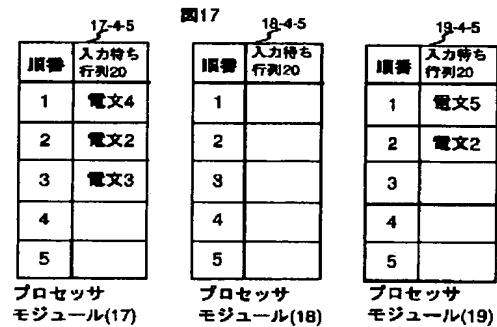


【図14】

図14

プロセッサモジュール	プロセッサ モジュールの アドレス
プロセッサモジュール(11)	(00)16
プロセッサモジュール(12)	(01)16
プロセッサモジュール(13)	(02)16
プロセッサモジュール(14)	(10)16
プロセッサモジュール(15)	(11)16
プロセッサモジュール(16)	(12)16
プロセッサモジュール(17)	(20)16
プロセッサモジュール(18)	(21)16
プロセッサモジュール(19)	(22)16
コンソール	(30)16

【図17】



【図18】

図18

電文種別	実行ステップ数		通信回数	
	ST1(i)	ST2(i)	CT1(i)	CT2(i)
電文1	100000	100000	4	1
電文2	150000	150000	5	1
電文3	130000	130000	3	1
電文4	130000	130000	2	1
電文5	170000	170000	3	1

【図20】

図20

プロセッサモジュール	処理能力 PT(i)	状 態
プロセッサモジュール(14)	50000000	(02)16
プロセッサモジュール(15)	40000000	(01)16
プロセッサモジュール(16)	40000000	(01)16

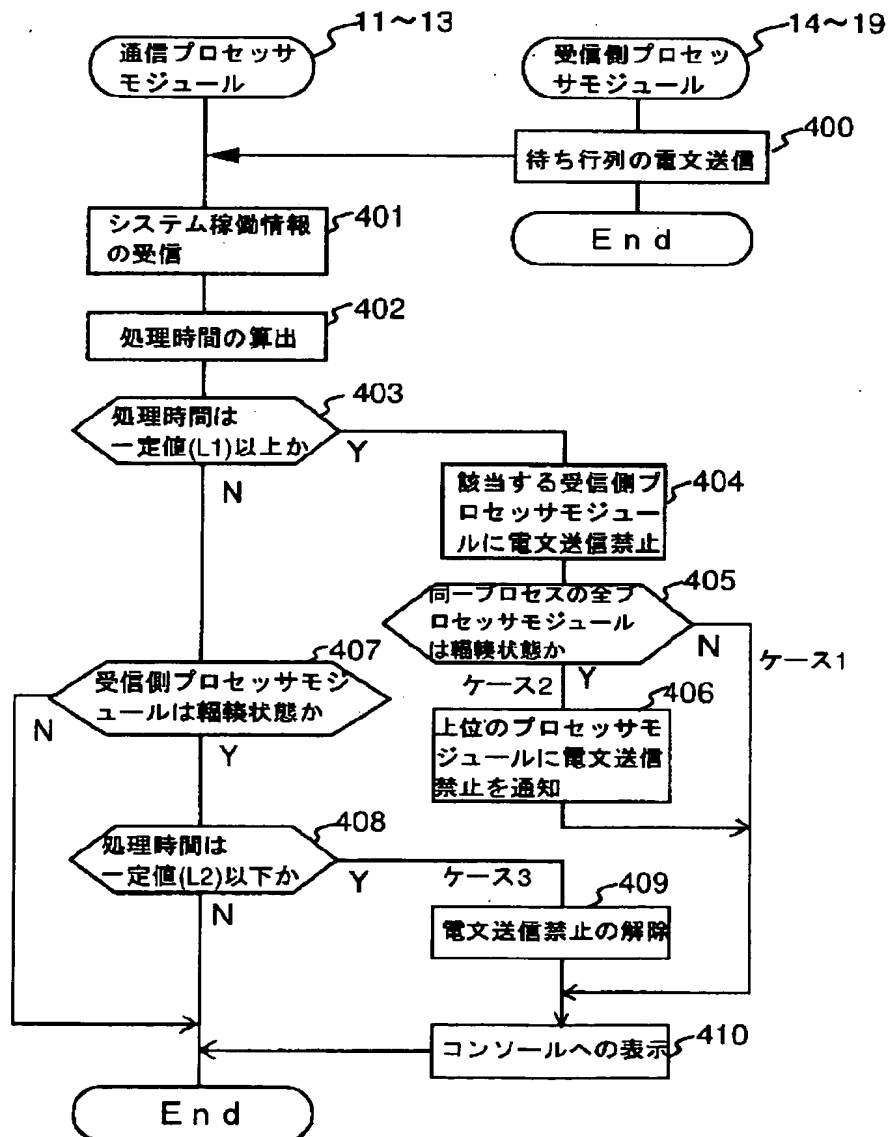
【図21】

図21

プロセッサモジュール	処理能力 PF(i)	状 態
プロセッサモジュール(17)	40000000	(01)16
プロセッサモジュール(18)	50000000	(03)16
プロセッサモジュール(19)	40000000	(01)16

【図22】

図22



【図23】

図23

(1)トランザクションプロセッサモジュール

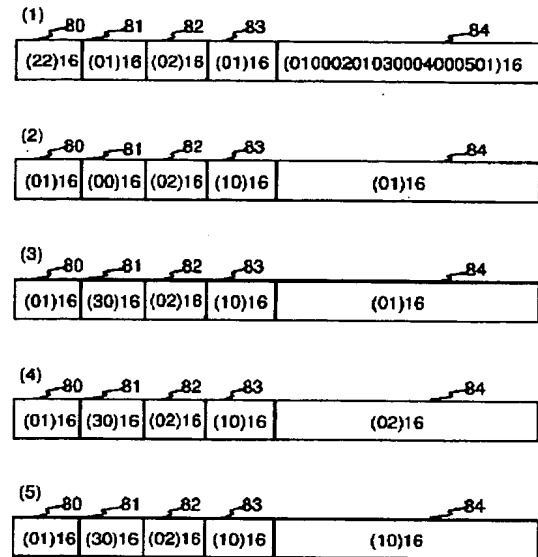
$$RT(i) = \frac{\sum_{j=1}^5 \{ST1(i) \times NT1(i) + ST2(i) \times NT2(i) + \{CT1(i) + CT2(i)\} \times PS\}}{PT(i)}$$

(2)ファイルプロセッサモジュール

$$RF(i) = \sum_{j=1}^5 \left(\frac{\{SF1(i) \times NF1(i) + CT1(i) \times PS\}}{PF(i)} + IF1(i) \times IT \right)$$

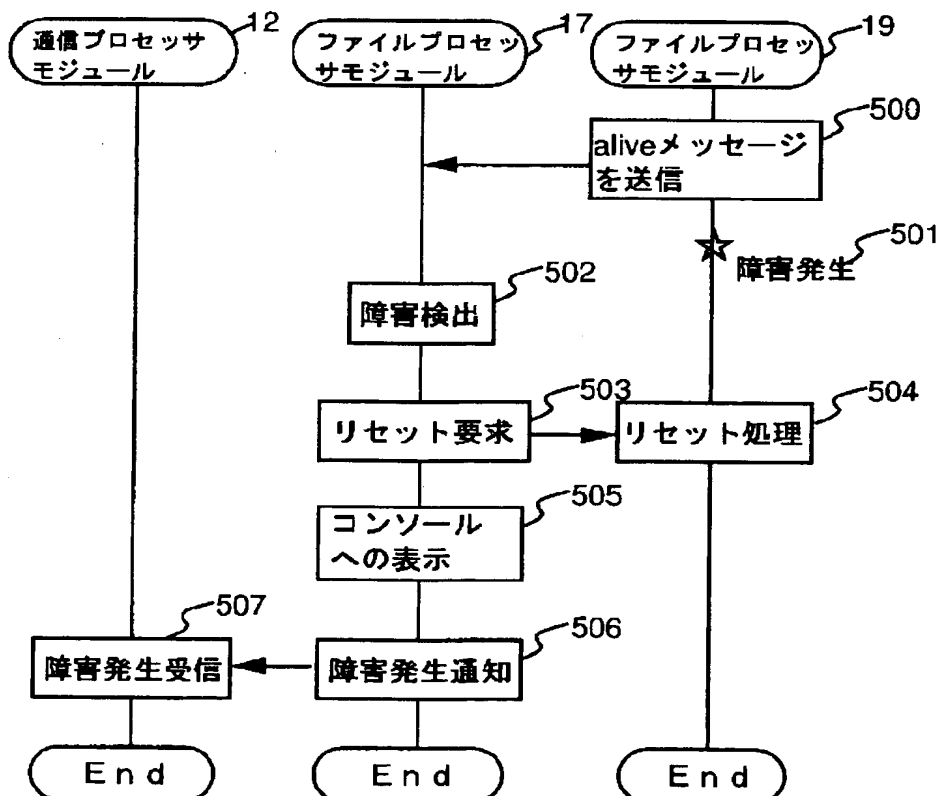
【図24】

図24



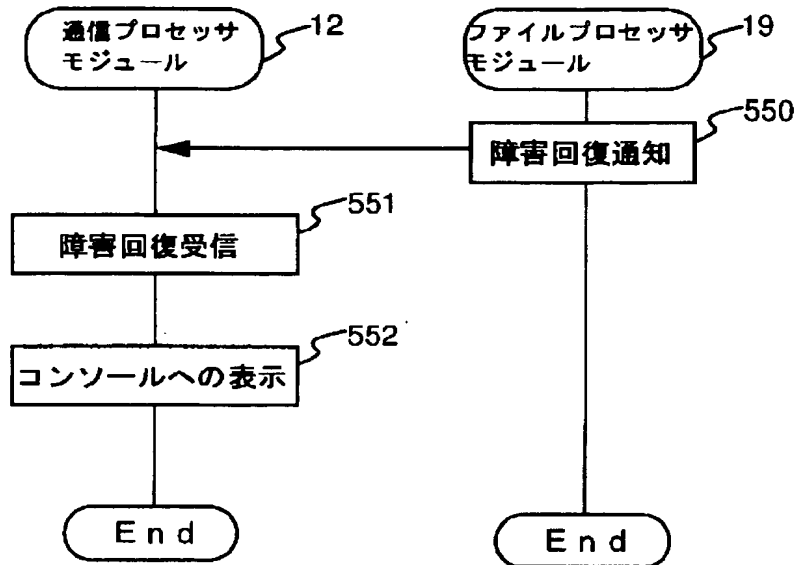
【図25】

図25



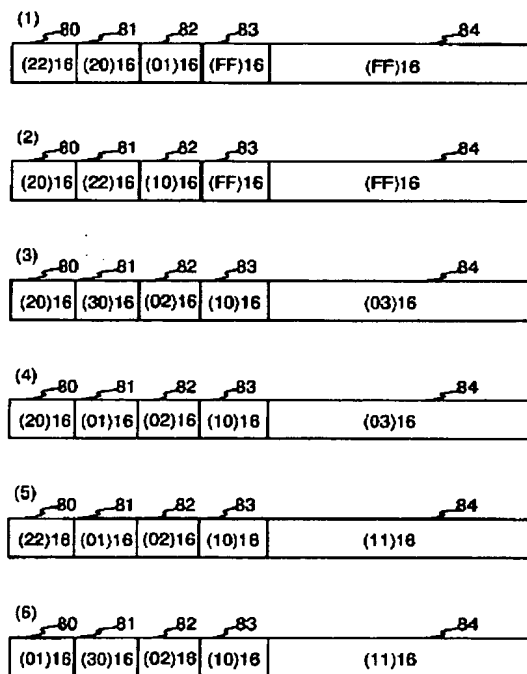
【図26】

図26



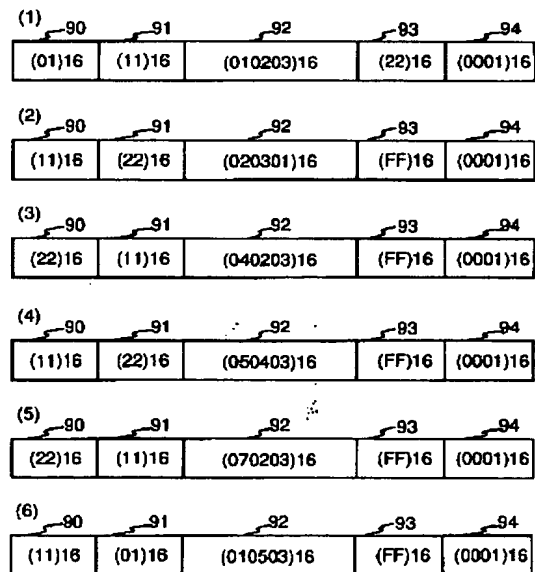
【図27】

図27

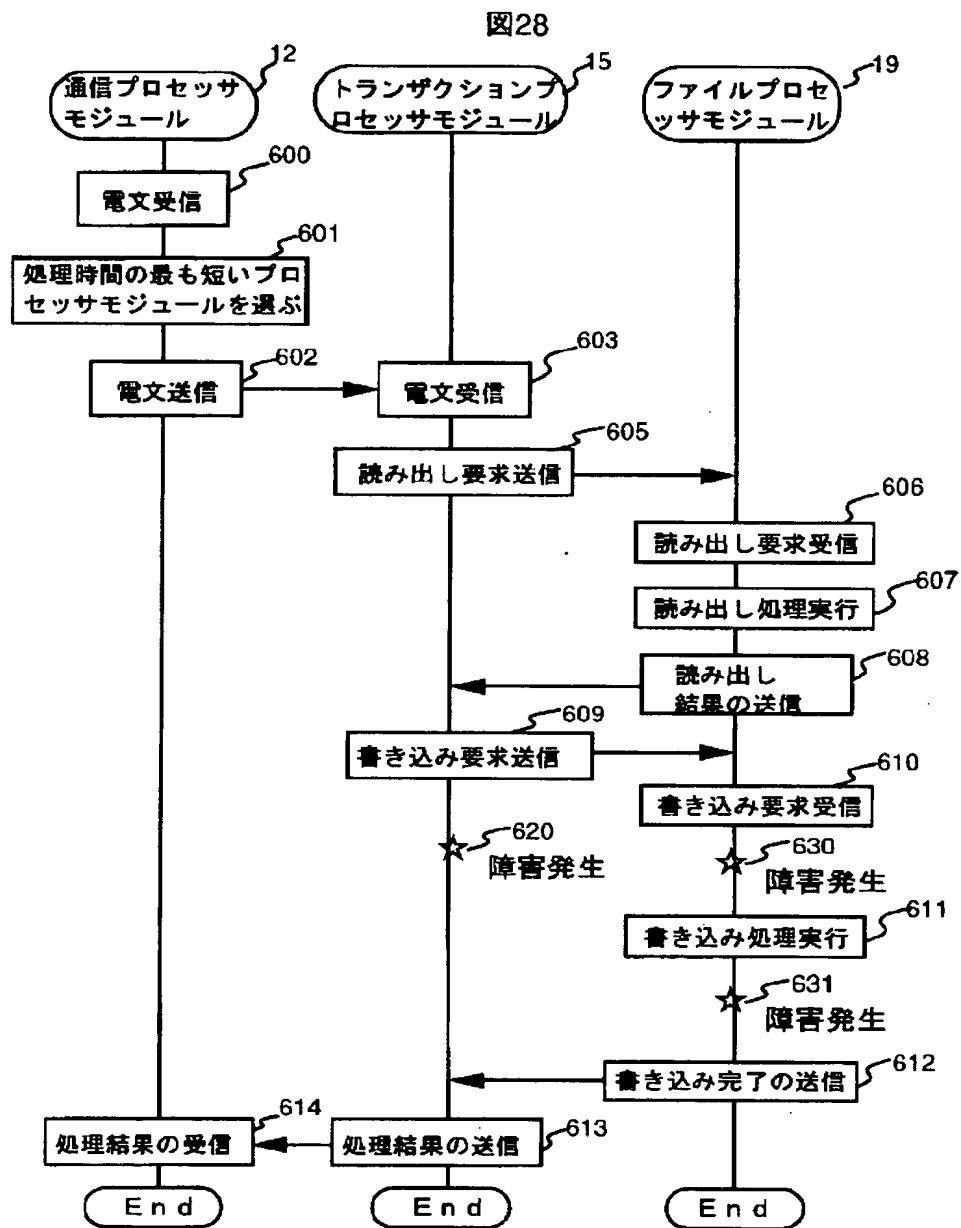


【図29】

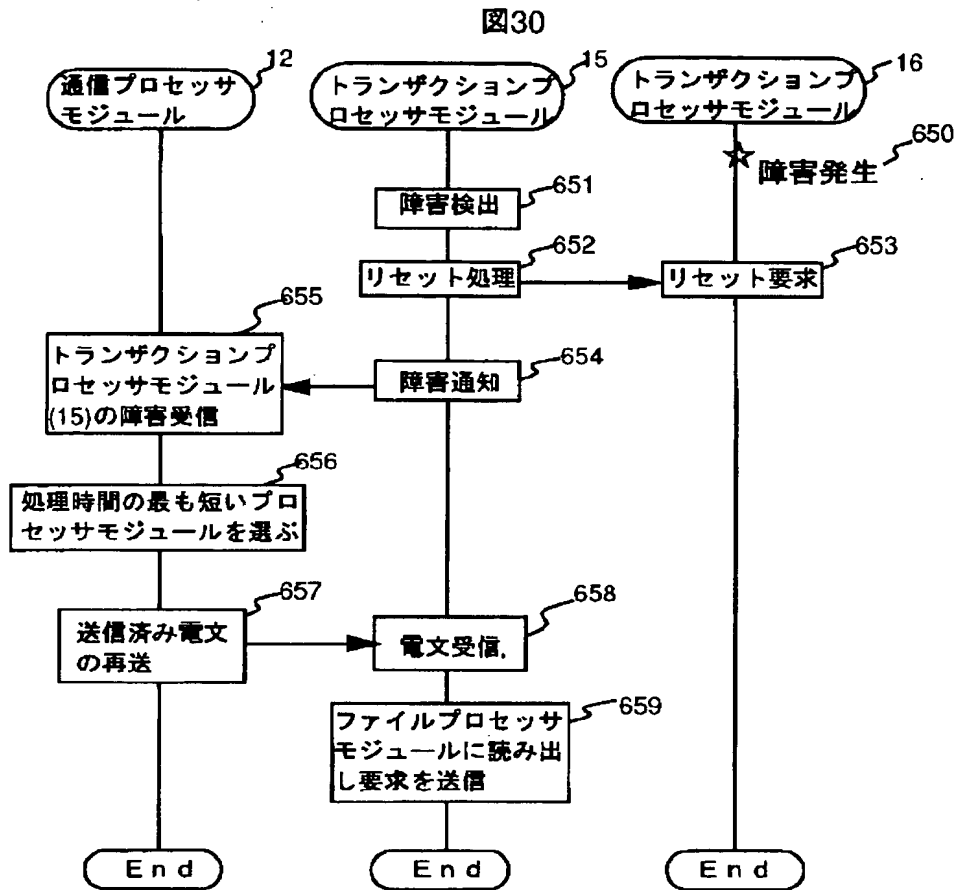
図29



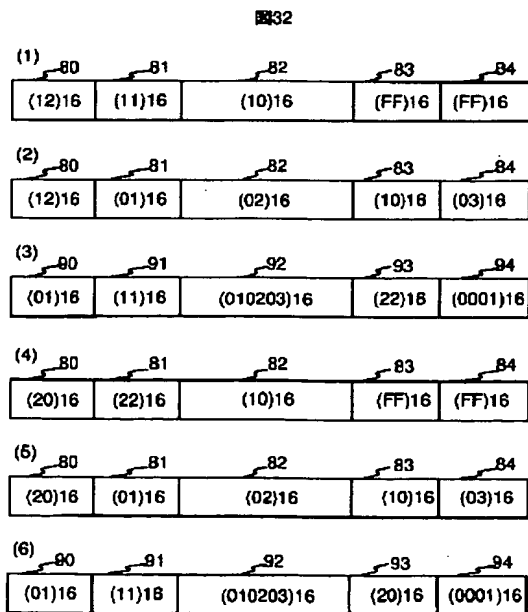
【図28】



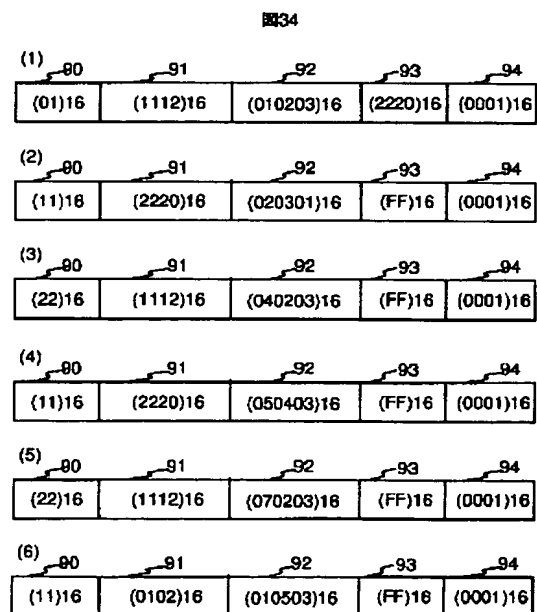
【図30】



【図32】

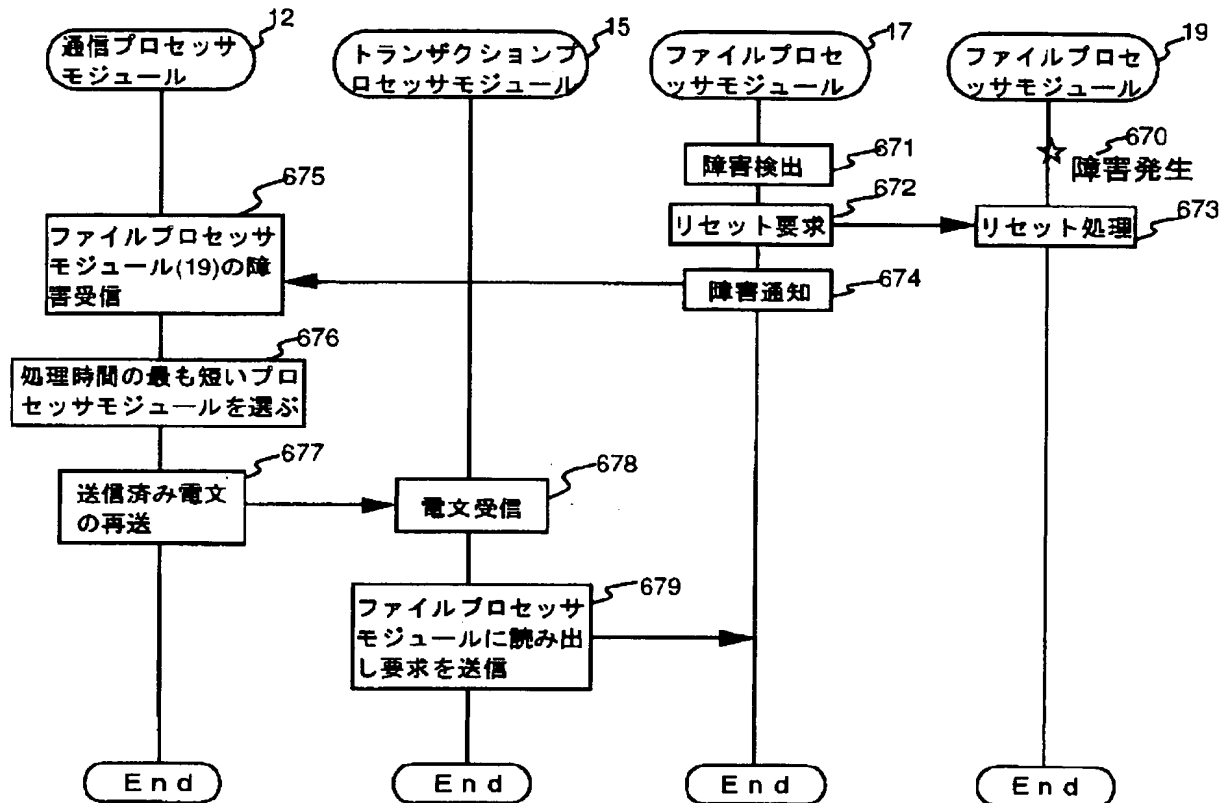


【図34】



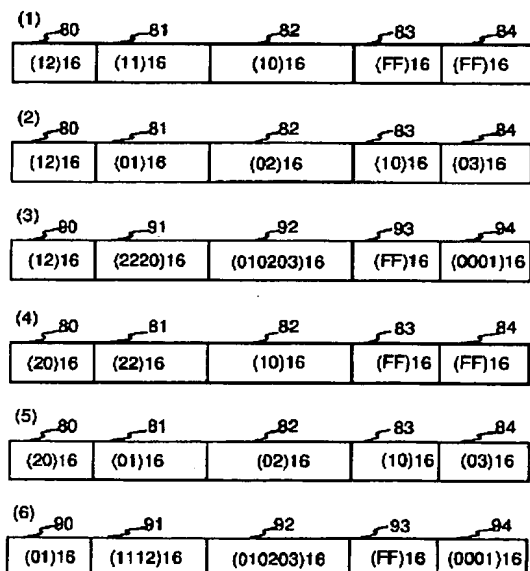
【図31】

図31



【図37】

図37



【図33】

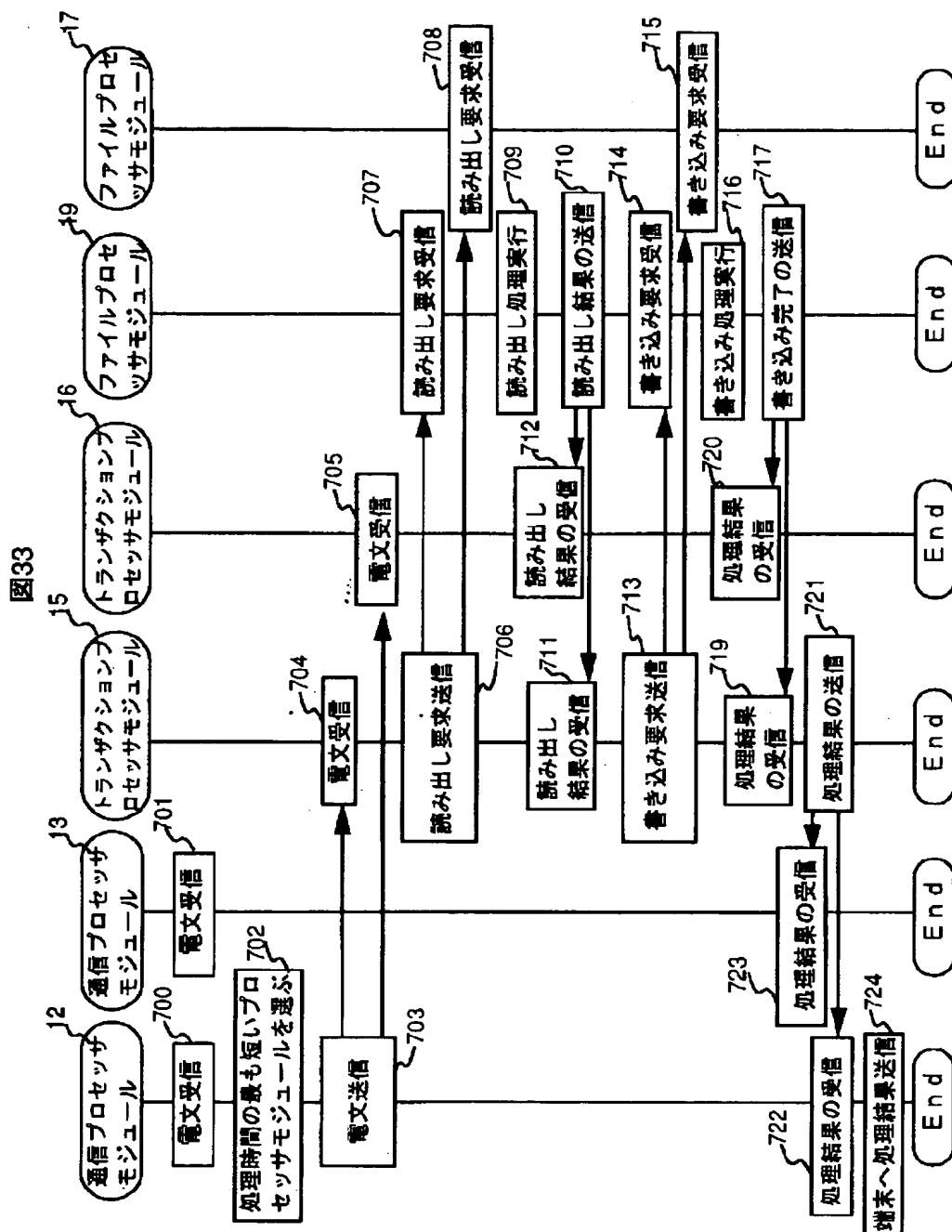
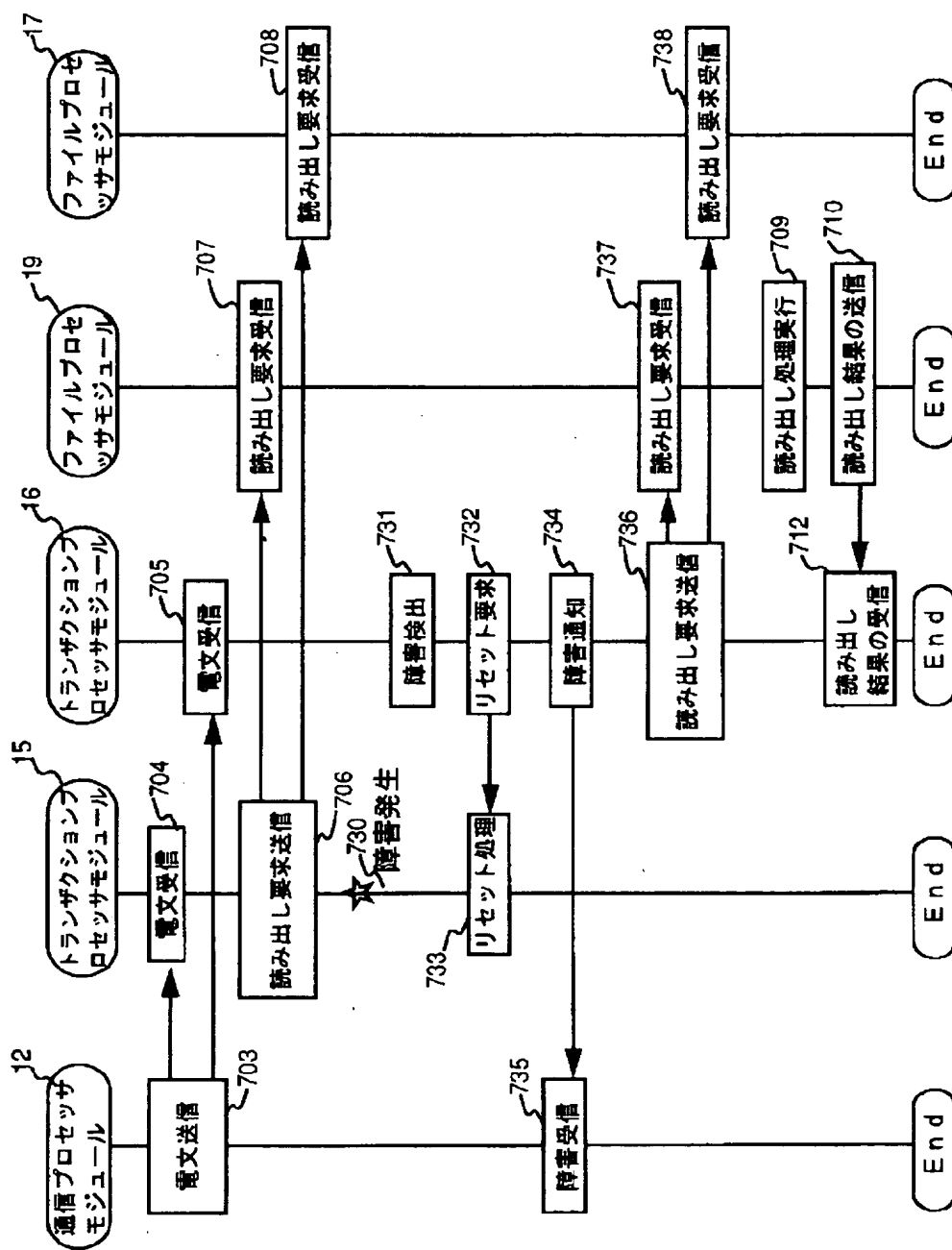


图 35



【図38】

